【Python安全攻防过渡篇：web编程、环境准备】

原创

白帽子续命指南 于 2020-10-29 21:38:23 发布 179 收藏 2

分类专栏： Python Web安全

本文链接：https://blog.csdn.net/qq_43428139/article/details/109371063

Python 同时被 2 个专栏收录

7 篇文章 1 订阅

订阅专栏

Web安全

18 篇文章 7 订阅

订阅专栏

web编程

web编程不是说用python做web开发，而是用python与web交互。常用的模块有`urlib,urlib2`，这是python内置的模块。

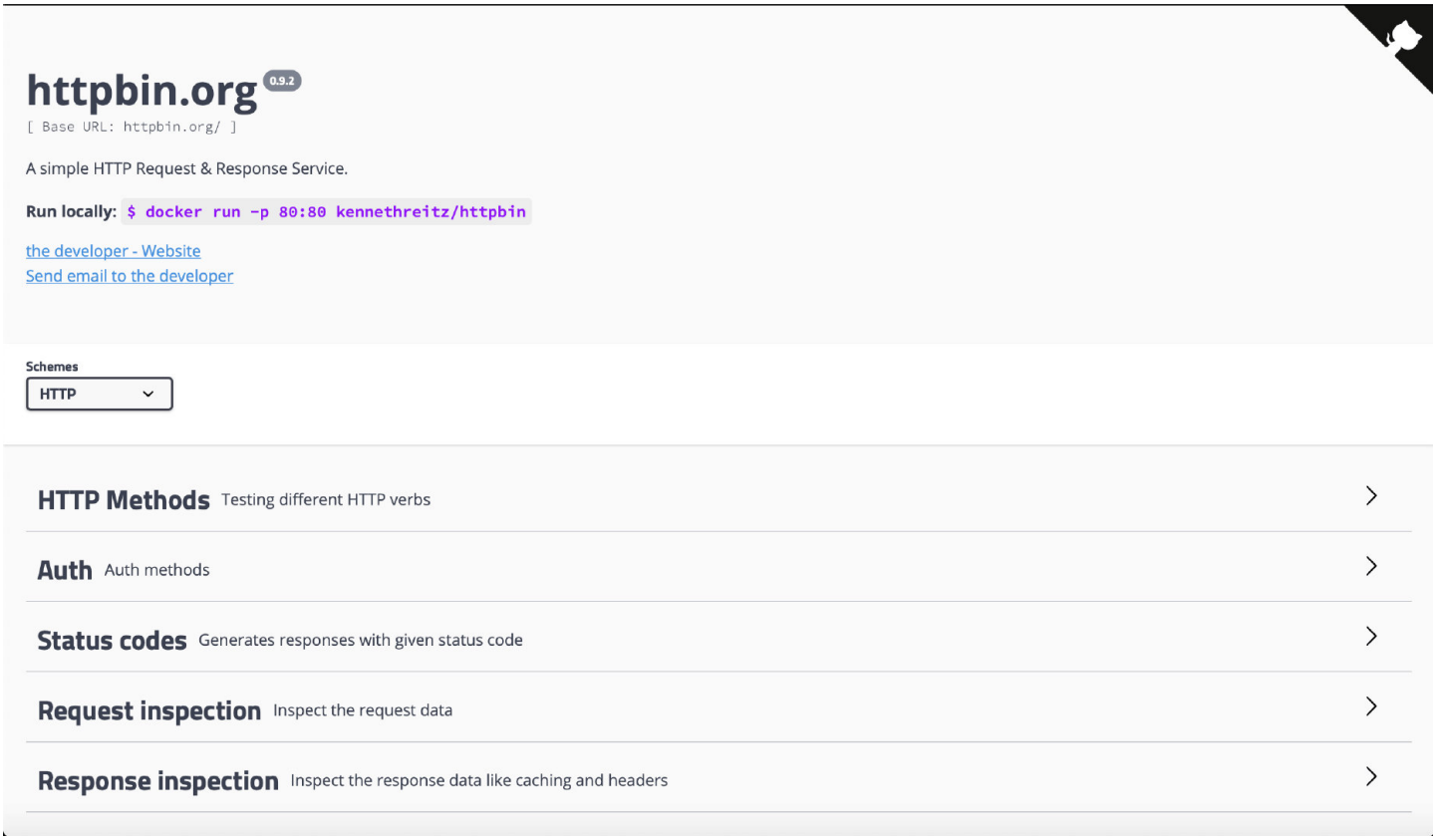
同时，还有基于urlib的第三方库，比如`requests, BeautifulSoup`，这里我们主要用`requests`举例，后期介绍爬虫的时候会详细说一下这些库/函数，**现在主要是带大家先了解web交互这一块，不然等会儿写代码没法写。**

准备条件

需要用到的测试网站

httpbin： http://httpbin.org

> A simple HTTP Request & Response Service.



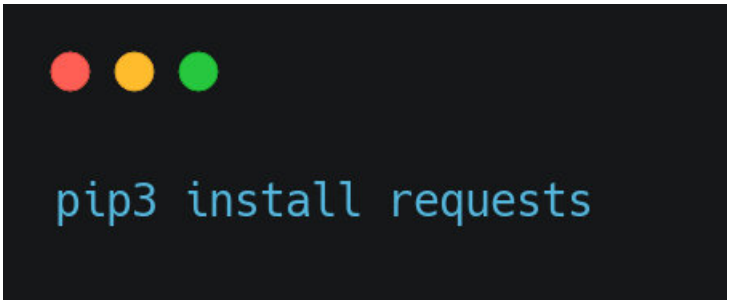
封神台在线靶场： https://hack.zkaq.cn/battle

> 测试靶场

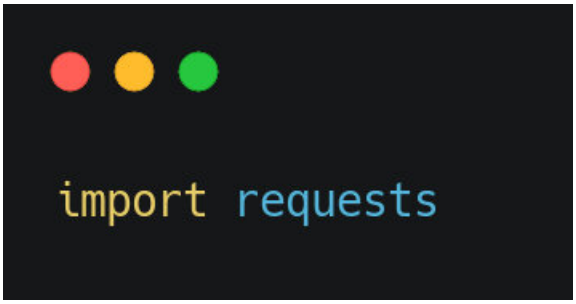
requests库安装

这是python的一个第三方库，用于处理URL资源。第三方，言外之意是python没有内置，需要我们手动安装。

安装



导入



基本请求方式

```
r = requests.get('http://httpbin.org')
r = requests.post('http://httpbin.org/post', data = {'key':'value'})
r = requests.put('http://httpbin.org/put', data = {'key':'value'})
r = requests.delete('http://httpbin.org/delete')
r = requests.head('http://httpbin.org/get')
r = requests.options('http://httpbin.org/get')
```

GET请求

最基本请求

```python
import requests

r = requests.get('http://httpbin.org')
print(r)
```

带参数的GET请求

params：向URL传参

```python
import requests

payload = {'id':'1'}    # 注意格式为字典
r = requests.get('http://httpbin.org', params = payload)
print(r)
```

headers：自定义请求头

```python
import requests

headers = {
    'User-Agent':'Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:81.0) Gecko/20100101
Firefox/81.0'
}
r = requests.get('http://httpbin.org', headers = headers)
print(r)
```

cookies：跳过登陆页面

还是用我们老朋友：`掌控安全旗下封神台靶场`做演示，使用携带cookies的方式跳过登陆。

```
import requests

cookies = {
    'Cookie':'UM_distinctid=1756f6dcaa59-03cae543ecf7148-445d6f-13c680-1756f6dcaa71d8;
CNZZDATA1274101648=547061955-1603892170-
https%252525253A%2F%2Fhack.zkaq.cn%2F%7C1603898895;
PHPSESSID=rm0g59j3b4oup6490no5e7k2up'
}

headers = {
    'User-Agent':'Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:81.0) Gecko/20100101
Firefox/81.0'
}
r = requests.get('https://hack.zkaq.cn/user/login', cookies = cookies, headers =
headers)

print('个人中心' in r.text)
```

1. 用burp或者cookie插件先获取cookie
2. 因为登陆成功右上角就会变成"个人中心"，所以用这个作为判断条件

如果上边这些代码执行结果全都是200/TURE的话，就证明你已经掌握GET请求了。废话不多说，看下POST请求。

一定要跟着敲一遍啊师傅们，不然越看越懵逼，而且马上就要自己写脚本了，连最基本的web请求都不会，真没法写！

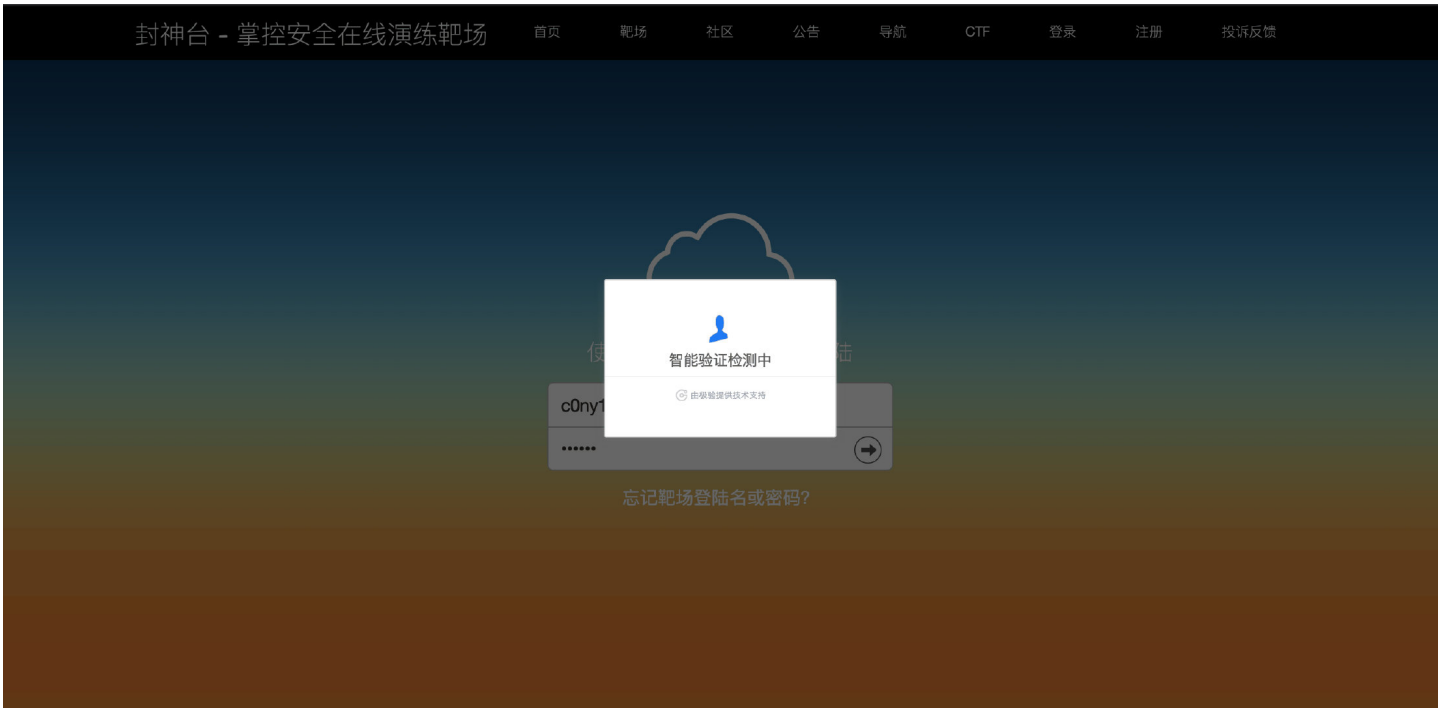
如何获取cookie

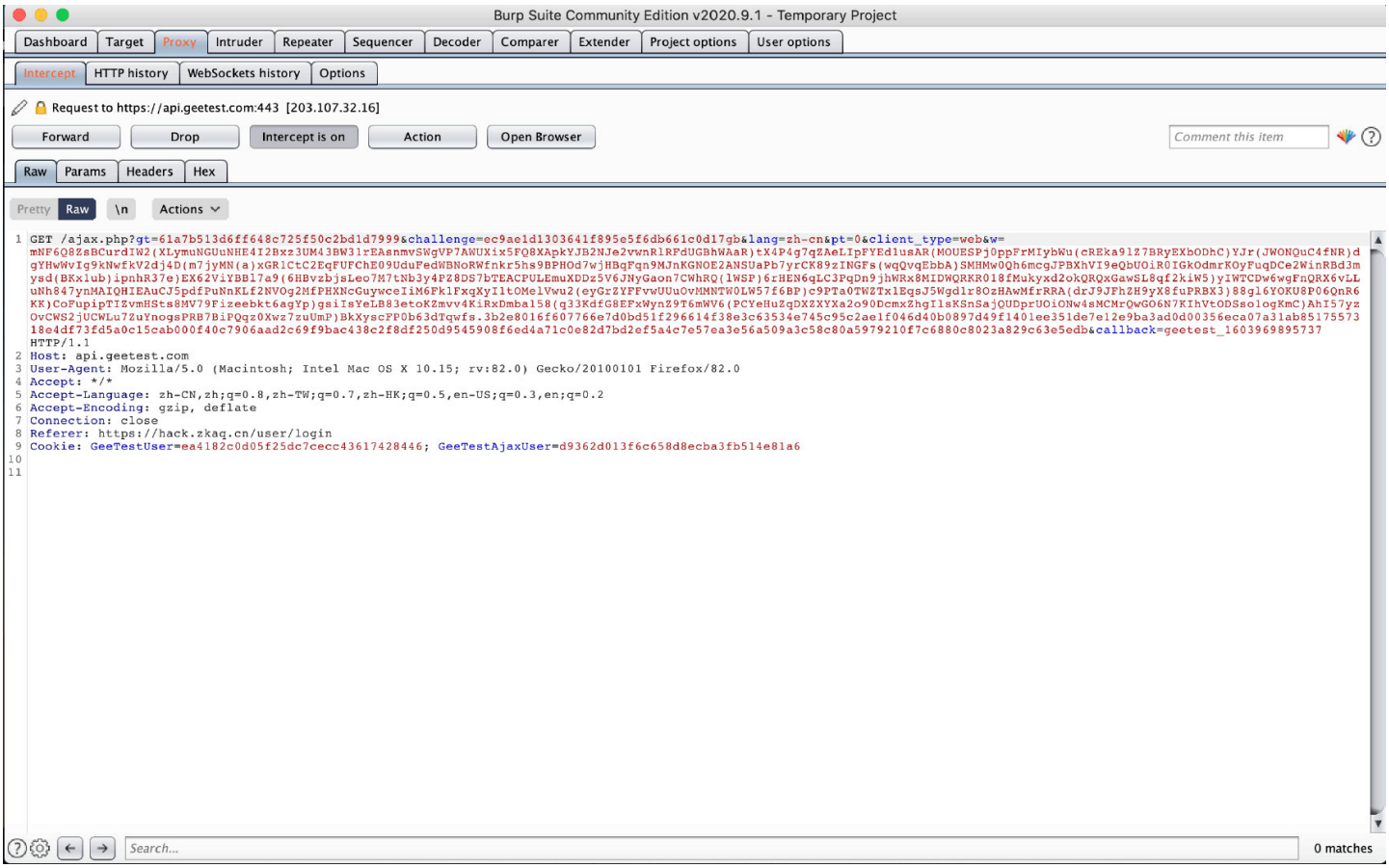
简述下获取cookie的过程，担心有些小白不太会，老师傅绕路：（这里以burp举例）

1、在这里点击登录

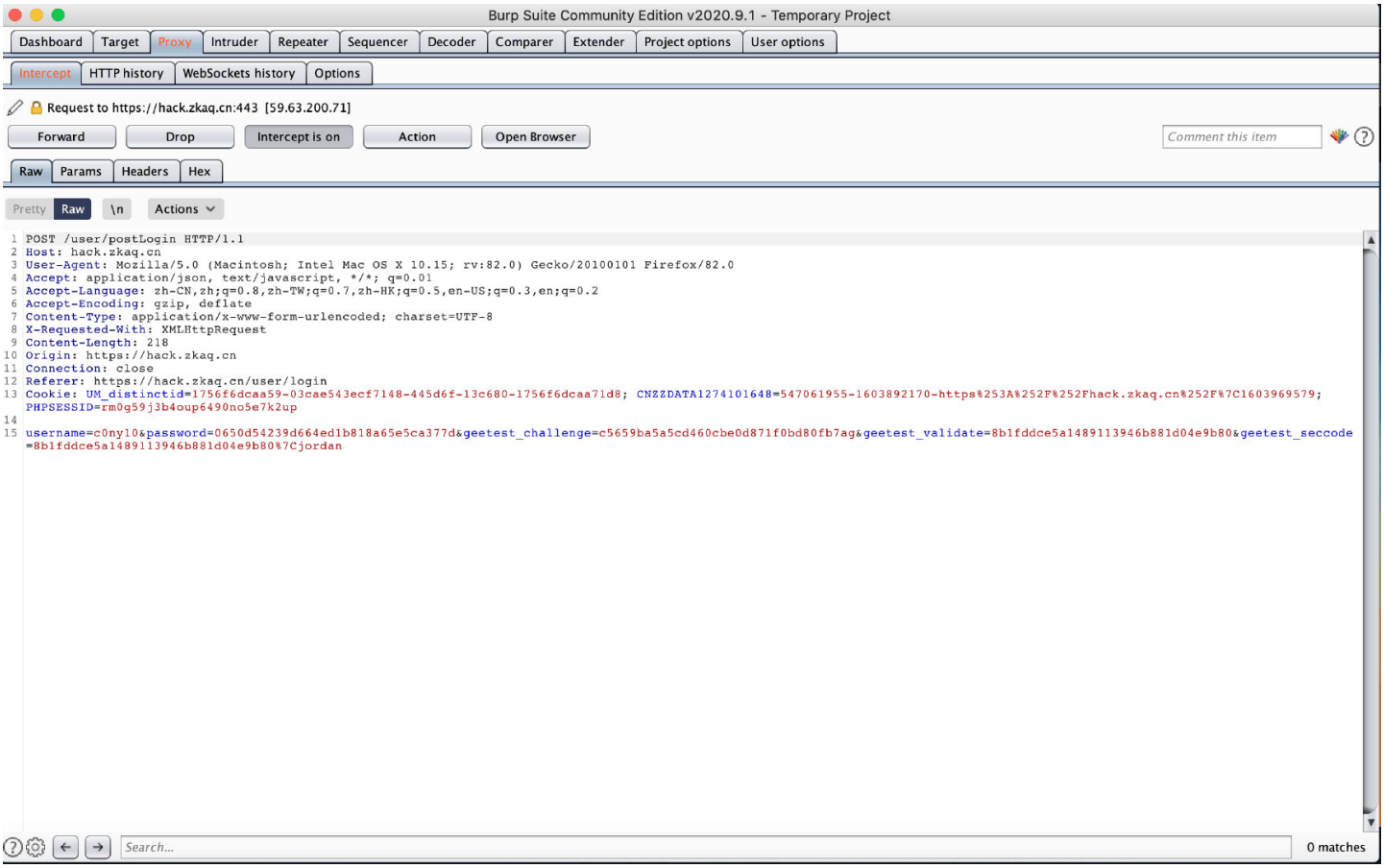
2、输入账号密码后，挂上burp代理，开启拦截，然后登录

3、这个时候因为burp的拦截开启着呢，所以需要我们手动点forward给数据包放出去





4、浏览器一边操作，burp一边forward，直到burp里有cookie出现



同时，这个时候复制cookie，或者保存数据包，后者等下去HTTP history里看都可以，但是这里要再点一下forward，看看这个cookie是不是能登录成功。

点一下forward出现这个界面，成了！

因为不知道朋友们基础都怎么样，反正我记着最开始看一些师傅的教程的时候，我就不知道怎么获取cookie，所以就简单提一下吧。

当然，python本身也是可以获取cookie的，但是再说多了我怕劝退，还是用咱们最熟悉的方式，先这么获取着吧。

Tips：平时做测试的时候，可以故意输错密码，向上边这样一点点分析请求过程。

POST请求

`POST请求和GET的用法完全一致，区别就是POST请求会带有一个data参数`

```python
import requests

# 普通的post请求
r1 = requests.post('http://httpbin.org/post')
# 高级的post请求
payload = {'name':'cony10', 'passwd':'12345'}
r2 = requests.post('http://httpbin.org/post', data = payload)

print(r1.text)
print(r2.text)
```

Response响应

```python
import requests

r = requests.get('http://httpbin.org/get')

print(r.status_code)    # 获取响应状态码
print(r.url)          # 获取url地址
print(r.text)         # 获取文本
print(r.content)          # 获取二进制流
print(r.headers)          # 获取页面请求头信息
print(r.history)          # 上一次跳转的地址
print(r.cookies)          # 获取cookies信息
print(r.cookies.get_dict())     # 获取cookies信息转换成字典
print(r.cookies.items())          # 获取cookies信息转换成字典
print(r.encoding)         # 字符编码
print(r.elapsed)          # 访问时间
```

环境准备

这部分主要是先搭建好后期可能会用的服务，主要就是配置下虚拟机，安装一些比如docker、web服务、ssh服务等等，方便后期使用。

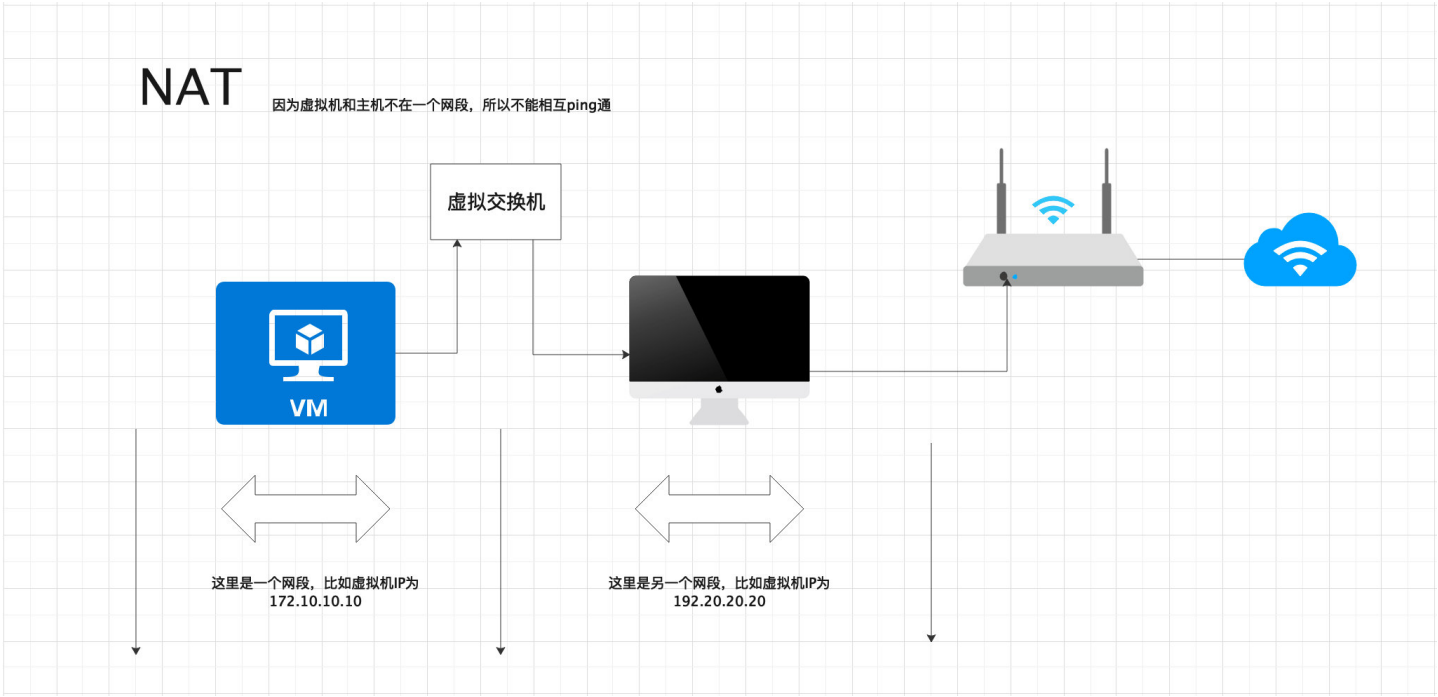
虚拟机配置

简单说下`桥接`和`NAT`的区别。

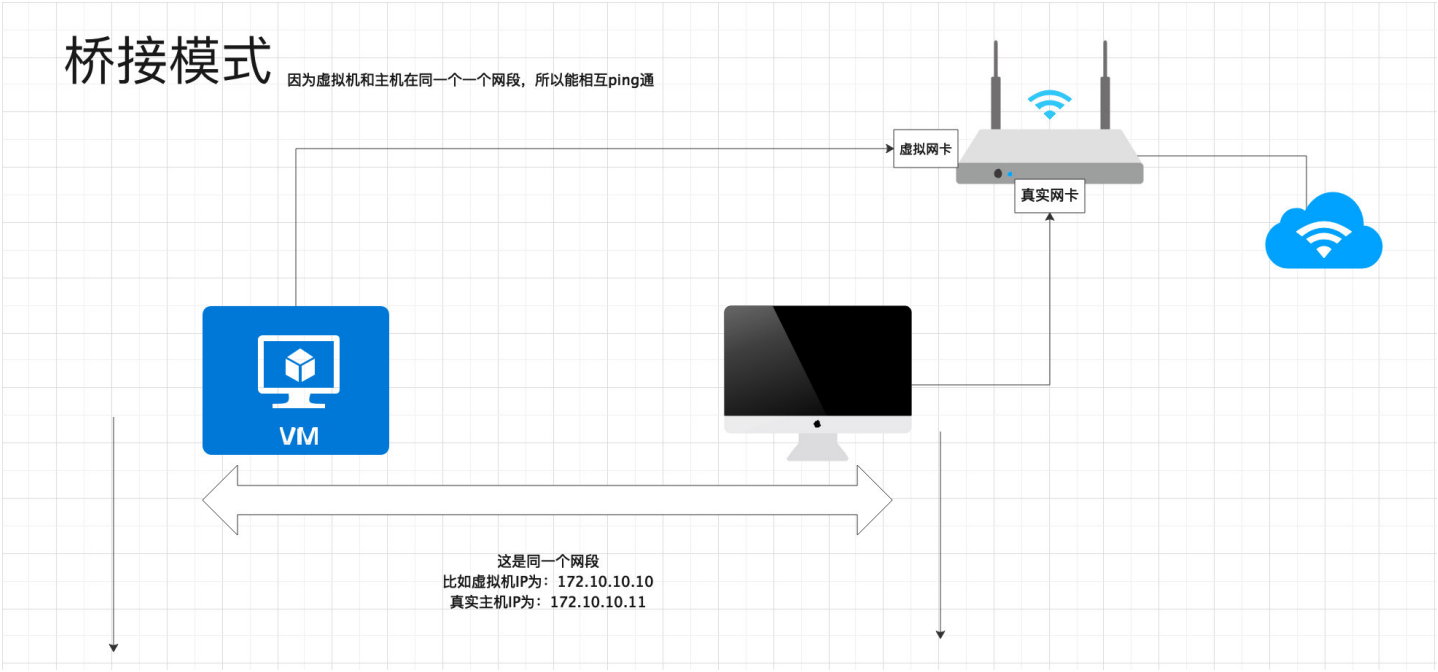
学过网络的应该知道NAT就是转发的意思，一般代理服务器都是基于这个功能，所以NAT的上网方式就是把本机当作代理服务器；桥接就是相当于把虚拟机通过无线网卡连在你当前网络的路由器上。

如果这段你听懂了就跳过吧，没听懂我详细说下：

NAT



桥接



所以我们需要把虚拟机配置成桥接模式，然后设置dhcp模式。具体自己百度吧，不同系统不太一样。

SSH

这东西是远程连接的一种，我们需要在**客户端有ssh连接工具，服务端有ssh服务**，就是这么简单点事。

服务端

只需要百度“centos安装ssh服务”、“Ubuntu安装ssh服务”就可以了。

确实需要一些配置操作，教程里一般都会提到的，放心。

PS：我直接下的CentOS，它直接ssh服务安装好，配置好了，各位如果还没安系统，可以考虑CentOS。

客户端

客户端我不推荐你们安装xshell什么的，太麻烦了。

1. 苹果电脑的话，用自带的Terminal；
2. Linux的话，用自带的Terminal；
3. Windows的话，~~用自带的Terminal~~，Windows自带的那个叫Dos是吧，哈哈，新出了个Terminal不会真有人不知道吧，去Microsoft商店搜索Terminal，下载一个就可以了。

Terminal干嘛用的？里边有ssh啊，直接在里边连不想吗？

> 这里给Windows用户一个建议，你连虚拟机都不用安，同样是微软商店搜索“Ubuntu”、“Kali”、“CentOS”，这些就是名为子系统的男人，在Terminal里能直接用，比虚拟机香多了。
> 当然了，具体的百度吧。这些网上教程太多了，写得也都很好，我就不想写了。关键字“Windows子系统”

连接

1. 去虚拟机看一下，输入`ps -e |grep ssh`，看看能不能输出带有`sshd`的字样，带d的就表示是服务端；
2. 还是虚拟机里，输入`ifconfig`看看IP是多少，比如我的就是`172.10.20.12`；
3. 去物理机，打开Terminal，输入`ssh root@172.10.20.12`，然后根据提示输入密码就完了。**密码就是你虚拟机的登录密码；**
4. 这样就不用每次都在虚拟机里操作了，爽吧？

---

docker

docker用于日后搭靶场，或者使用docker安装其他东西，都要快一些。**跟着我一起操作，就在Terminal里，谁都不许回虚拟机看图形界面！**

安装

我是CentOS7，可以使用国内的daocloud安，命令为：

```
curl -sSL https://get.daocloud.io/docker | sh
```

然后等它安完就可以了。其余系统百度吧兄弟们哈哈，这种东西我真的不想写，因为教程太多，其他人写的也都太好了！

比如菜鸟教程：https://www.runoob.com/docker/docker-tutorial.html

使用

比如后期想要安一个sqli-labs，直接一条命令：

```
docker pull c0ny1/sqli-labs
```

然后run一下就完了。以后用到会介绍，先不用着急哈。

Web服务

无非就是数据库、中间件、后端环境，通常为MySQL、Apache、PHP嘛。

两种安装方式你们选：

1. 直接嫖一个建站系统，我习惯用宝塔，这种可以一键安装所有服务；
2. 挨个安装！

我选直接宝塔CMS了：

> 使用 SSH 连接工具连接到您的 Linux 服务器后，根据系统执行相应命令开始安装（大约2分钟完成面板安装）：
> Centos安装脚本
>
> ```
> yum install -y wget && wget -O install.sh http://download.bt.cn/install/install_6.0.sh && sh install.sh
> ```
>
> Ubuntu/Deepin安装脚本
>
> ```
> wget -O install.sh http://download.bt.cn/install/install-ubuntu_6.0.sh && sudo bash install.sh
> ```
>
> Debian安装脚本
>
> ```
> wget -O install.sh http://download.bt.cn/install/install-ubuntu_6.0.sh && bash install.sh
> ```
>
> Fedora安装脚本
>
> ```
> wget -O install.sh http://download.bt.cn/install/install_6.0.sh && bash install.sh
> ```

注意：必须为没装过其它环境如Apache/Nginx/php/MySQL的新系统,推荐使用centos 8.X的系统安装宝塔面板

这样网站需要的服务我们就搭完了，以后可以自己搭测试环境了。这里有几点说一下吧，我之前刚入门的时候特别不懂的地方：

1. LAMP是什么？L(Linux), A(Apache), M(MySQL), P(PHP)；
2. 没有这四(三)个东西，能搭网站吗？不能！
3. 不做交互，只是一个静态web，需要全部安装吗？不需要，安一个Apache或者Nginx之类的就行了。

写在最后

马上就要开始自己动手写脚本，不知道师傅们心情怎么样？

反正我是蛮激动的，我打算自己做一个渗透测试网站，后台就用python写，大家有兴趣和我一起吗？

其实我也是python小白，直接也没用python搞过安全这一块，大家不用觉得用压力。

看这个教程，只需要你掌握python基础语法，然后熟悉渗透测试的常见漏洞就可以了。

我遇到高阶操作也是看网课看教程这样子，并且我都会在文章中说的，大家不要害怕python安全。

就比如这一篇，就是我在学后边内容发现requests库需要学一波，但是学完又发现，貌似又用不到那么多，只是会简单的请求和响应就可以了。

所以就出了这一个过渡篇。

同时又看到后边很多测试可能需要自己搭环境，比如漏洞检测，就需要自己搭一个靶场或者自己写一个网站，比如绕狗，就需要自己安一个狗，这样子。

总之就是告诉大家，我不是python大佬，我不会降维打击的，大家不要怕，不敢写，这样永远都不可能进步了。

References

[1] https://www.cnblogs.com/Booker808-java/p/7822763.html

[2] https://blog.csdn.net/Eastmount/article/details/108540749

[3] https://blog.csdn.net/nilvya/article/details/103674999

[4] https://www.cnblogs.com/kermitjam/p/10863916.html#_label3