目录

网络中转站：代理技术

- 最新版本：`v1.2`
- 更新时间：`20230824`

简介

总结应用广泛的网络中转技术，代理技术。总结代理的主要用途，相关技术和工具。以及如何获取到代理，包括爬虫代理、抓包代理、科学上网代理等。以及爬虫代理中的购买代理相关的基础知识，比如代理池、私密代理vs隧道代理、扫描vs非扫描、是否去重、多个请求锁定一个IP、每秒请求次数、认证方式、代理IP检测、免费试用等，以及常见代理服务商公司，比如阿布云代理、多贝云代理等。然后介绍抓包代理，和安装根证书等基础知识。总结如何添加代理，包括各种编程语言，尤其是Python中的requests等，和常见开发工具中，比如git、pip、Android Studio、gradle、rsync、SecureCRT等，以及移动端的iOS和Android，桌面端的Mac和Windows等等。

源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

HonKit源码

- crifan/web_transfer_proxy_tech: 网络中转站：代理技术

如何使用此HonKit源码去生成发布为电子书

详见：crifan/honkit_template: demo how to use crifan honkit template and demo

在线浏览

- 网络中转站：代理技术 book.crifan.org
- 网络中转站：代理技术 crifan.github.io

离线下载阅读

- 网络中转站：代理技术 PDF
- 网络中转站：代理技术 ePub
- 网络中转站：代理技术 Mobi

版权和用途说明

鸣谢

感谢我的老婆**陈雪**的包容理解和悉心照料，才使得我 `crifan` 有更多精力去专注技术专研和整理归纳出这些电子书和技术教程，特此鸣谢。

其他

作者的其他电子书

本人 `crifan` 还写了其他 `150+` 本电子书教程，感兴趣可移步至：

crifan/crifan_ebook_readme: Crifan的电子书的使用说明

关于作者

关于作者更多介绍，详见：

关于CrifanLi李茂 – 在路上

代理概述

- 代理
  - 英文：`proxy`
  - 中文
    - 直译：代（替你去处）理
    - 含义：通过一个中转站去处理某些事情
- 思考：为何需要中转站?
  - 答：实现某些目的
    - 提高（下载文件等）速度
    - （作为中国人）访问（没有经过中转就无法打开的）国外的技术网站
    - 其他方面用途

代理主要用途

- 代理的主要用途=典型用途
  - 爬虫领域
    - 反反扒
      - 模拟不同IP多人同时访问
  - 破解领域
    - 安卓
      - 绕过https证书检测
  - 科学上网(翻墙)领域
    - 访问被禁国外技术网站
    - 加速

代理相关技术和工具

对于不同用途的代理，对应的所用技术和工具是：

- 爬虫
  - 涉及：
    - 服务
      - 免费或收费的代理
        - 免费
          - 网上找免费可用代理
        - 收费
          - 自己从某代理服务（卖代理的网站）中购买对应代理（服务）
            - 比如
              - 阿布云代理
              - 多贝云代理
- (app)抓包
  - 涉及：
    - 工具
      - `Charles`
        - app抓包利器：Charles
      - mitmproxy
        - 抓包代理利器：mitmproxy
    - wireshark
- 科学上网(翻墙)
  - 涉及
    - 技术
      - Shadowsocks
      - V2Ray
      - Trojan
    - 工具
      - 对应客户端
        - ShadowsocksX-NG
        - V2rayU
        - Trojan-QT5

代理协议种类

不论是哪种代理，都涉及到代理的协议种类：

- 常见代理协议种类
  - `http`
  - `https`
  - `socks`
    - 协议
      - `SOCKS4`
      - `SOCKS5`
    - `SOCKS` 代理
      - 只是单纯传递数据包，不关心具体协议和用法，所以速度快很多
      - 端口一般为：`1080`

如何获取到代理

想要用代理之前，先要得到代理，即得到代理的配置信息，主要是代理服务器的

- IP地址或域名
- 端口
- 协议种类

如前所述，代理根据用途主要有：

- 爬虫类
- (app)抓包类
- 科学上网类

下面分别详细介绍，对应的获取代理的方式。

爬虫代理

爬虫开发期间，往往为了实现反扒等需求而需要用到代理。

可以网上找免费的代理，也可以购买收费的质量更高的代理。

购买代理

如果做爬虫，可以考虑通过买别人的代理，实现动态IP等功能，实现反扒效果。

目前市场上有多家代理公司，卖各种代理服务。

基础知识

此处购买代理服务之前，需要了解的一些常用的基础知识。

IP代理池

IP代理池，顾名思义，就像一个水池，，其中放了很多可用的代理IP地址，供需要时取用。

此处往往指的是，你所购买的代理服务、代理套餐，在提供随机动态的IP时，就像有个池子一样，有大量的IP放在里面可供选用，所以叫做：`IP池` = `代理池` = `IP代理池`

所以此时不同提供商会有个指标叫做：IP池的量

比如：

- 套餐购买-芝麻软件
  - 每日40万IP
- 蘑菇代理 - 购买API代理
  - 每日15万高匿名IP

当然，相对来说，代理池的IP个数越多，使用起来越好，越容易起到反扒效果。

与之相关，还会有切换IP的策略：

比如：

- 专业版HTTP隧道服务说明 | 阿布云 - 为大数据赋能
  - 如果全局 IP池 中有当前隧道当天未被使用过的 IP，则从未使用过的 IP 中随机挑选一个；
  - 如果全局 IP池 中的 IP 都曾被使用过，则从全局 IP池 中随机挑选 IP。
  - 不同隧道可能会拿到相同的 IP，如果使用者的应用对重复 IP 敏感，需要在应用中自行做 IP 过滤。

私密代理 vs 隧道代理

在一些IP代理池提供商中看到了：

- 私密代理IP
- 隧道代理IP

比如：

- 蜻蜓代理 - 企业级高质量代理ip平台
  -
- 蘑菇代理 - 购买API代理
  -
- 动态版HTTP隧道服务说明 | 阿布云 - 为大数据赋能
  - 动态版 HTTP隧道

好像没太明白：

IP代理池，都是动态的IP

但是 私密 vs 隧道 是什么区别？

再自己看解释：看懂了：

- 蜻蜓代理
  - 文档
    - 购买私密代理 - 蜻蜓代理 - 企业级高质量代理ip平台
    - 购买隧道代理 - 蜻蜓代理 - 企业级高质量代理ip平台
  - 含义

        - 私密代理
            - 通过API提取IP
                - 全部代理高匿名，白名单地址绑定，每天最高40万IP
        - 隧道代理
            - 统一入口，随机动态IP
                - 专为爬虫采集业务而定制
- 蘑菇代理
    - 文档
        - 购买 蘑菇代理 - 购买API代理
        - HTTP 蘑菇代理 - 购买API代理
        - 帮助信息 蘑菇代理 - 购买API代理
    - 含义
        - 私密API代理
            - 通过API提取IP
            - 请求API接口，直接返回IP与端口
        - HTTP隧道代理(动态转发)
            - 统一入口，随机动态出口，每一个请求一个随机IP
            - 接入固定代理服务器，动态转发请求

总结：

- 私密代理=私密API代理=私密代理API
    - 获取动态IP方式：通过API获取代理服务器的信息（IP和端口）
    - 使用方式：你调用api接口获取到IP和端口，自己用IP和端口去代理试用
        - 后来也才看懂
            - 大象代理IP提取-IP地址购买-API-代理服务器提取购买
                - 中的：
                    - API返回格式：文本
                    - API是否返回地区
                    - API是否返回运营商
            - 购买代理 - 快代理
                - 中的：
                    - API最快调用频率 1秒10次
                    - API允许调用IP数 2个起
                    - API支持返回json, xml
            - 短效优质代理IP- 站大爷
                - 中的：
                    - 每次提取IP数：5个
            - 说的就是：
                - 你自己：去调用api
            - 返回
                - 的内容：IP地址列表
                    - 即一次性往往会返回多个IP地址
                        - 比如 `5` 个， `10` 个， `100` 个等等
                - 格式： `xml` / `json` / `txt`
            - 且你调用API接口的频率也不能太高
                - 比如最多 `10次/秒`
- HTTP隧道代理
    - 获取动态IP方式：通过http代理提取
        - 此处的 http代理=HTTP隧道=HTTP隧道代理=服务商提供的一个HTTP代理服务器
            - 可以动态的，根据不同请求，内部使用不同IP
    - 使用方法：配置好HTTP（隧道）代理服务器后，后续请求，先到HTTP代理服务器，内部会自动的使用动态的IP转发请求
        - 适合爬虫采集

HTTP隧道代理

IP代理 根据协议，分很多种：

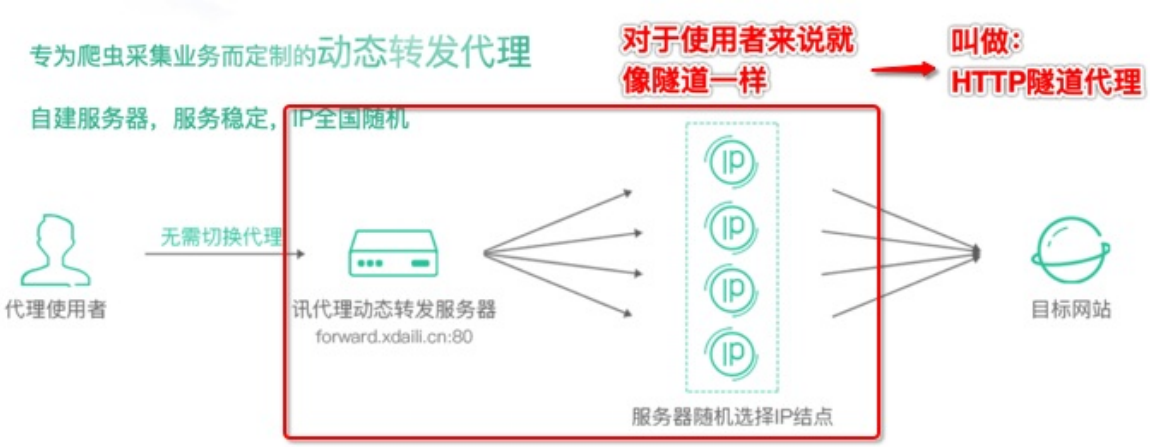
- HTTP/HTTPS
- SOCKS
- 等等

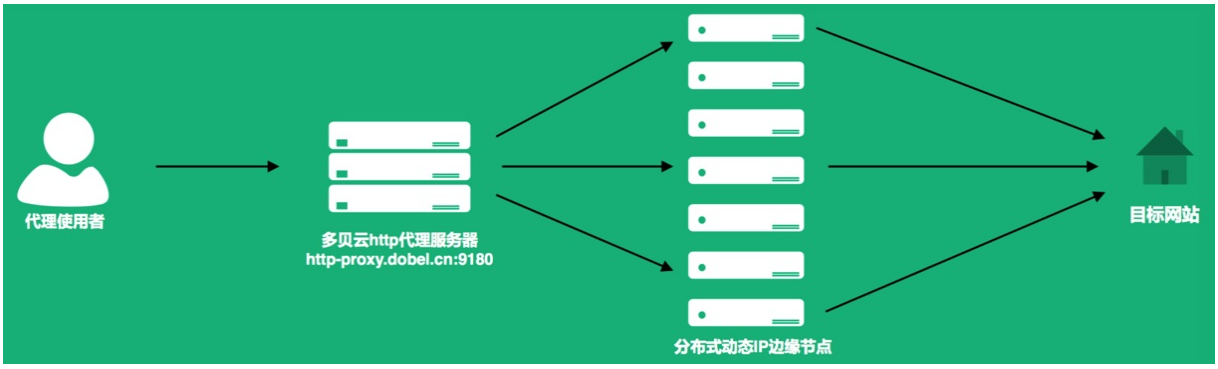
其中用HTTP协议的隧道代理，叫 `HTTP隧道代理`

比如：

- HTTP隧道产品 | 阿布云 - 为大数据赋能
- 讯代理-爬虫代理-HTTP代理-代理服务器
- 多贝云|动态IP代理拨号代理VPN代理动态HTTP代理

用图解释就更容易懂了：

- 

- 

扫描vs非扫描

关于代理的IP的来源：

- 扫描的
  - 很多都是免费的，不要钱的
  - 所以往往是从网络上，通过扫描，搜集得到的
  - 所以**质量和稳定性一般不是很高**
- 非扫描的
  - 所以很多收费的，质量好的
    - 都不是从网上扫描得到的
    - 而是自建节点的
    - 所以**稳定**、**速度**等**质量**都相对**更好**些

是否去重

对于有些业务，可能会要去发出的请求，不能有重复的IP

对此，IP代理提供商，有的确保IP不重复，有的不确保，需要你自己过滤和去重。

比如：

- 套餐购买 - 芝麻HTTP代理
  - 永久去重
    - 永远不会用到重复的IP

就不需要自己再去去重。

多个请求锁定一个IP

动态版HTTP隧道服务说明 | 阿布云 - 为大数据赋能

> 产品说明
>
> HTTP隧道 基于 HTTP 协议，支持 HTTP/HTTPS 协议的数据接入
>
> 平台在云端维护一个全局 IP池 供 HTTP隧道 使用，池中的 IP 会不间断更新，以保证 IP池 中有足够多的 IP 供用户使用。
>
> 需要注意的是 IP池 中有部分 IP 可能会在当天重复出现多次。
>
> 动态版 HTTP隧道 会为每个请求从 IP池 中挑选一个随机 IP。

然后再仔细看，才懂：

HTTP隧道产品 | 阿布云 - 为大数据赋能

- 每个请求一个随机IP
  - 每次http请求，IP都不同
- 多个请求锁定一个IP
  - 每几次连续的IP池的请求，或者说每一段时间内的请求，IP是一样的
    - 典型的是，对应的是：
      - 比如
        - 专业版HTTP隧道服务说明 | 阿布云 - 为大数据赋能

          > 每一个 IP 从隧道切换至该 IP 开始计时，最多只能使用 1 分钟，到期后系统将强制切换到另一个 IP

        - 就是说：每1分钟内的请求，都是同一个IP
          - 更深入的解释相关内容

            > 两次手动切换 IP 的间隔时间不得少于 1 秒

          - 指的是：如果支持手动切换IP（某些套餐才支持，有些不支持）的话，（调用api去）切换IP时，

不能太频繁，间隔不能小于1秒

- 但是对于更多的请求来说，IP是变化的

每秒请求次数

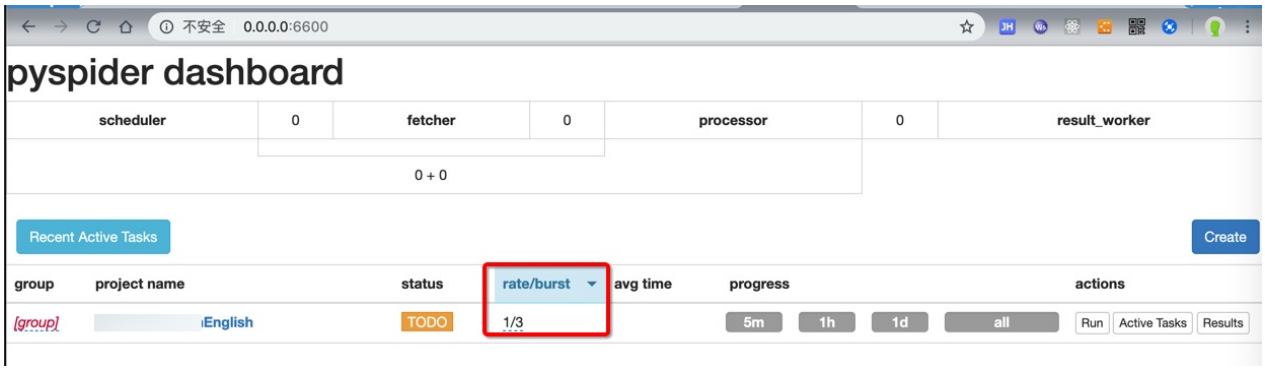
很多代理提供商（的不同套餐）都有提到：每秒请求次数 即：每秒的请求，不能超过几次

比如：

- 动态版HTTP隧道服务说明 | 阿布云 - 为大数据赋能
  - 动态版 HTTP隧道 有并发请求限制，默认每秒只允许 5 个请求
    - 如果需要更多请求数，请额外购买
- 多贝云|动态IP代理拨号代理VPN代理动态HTTP代理
  - 单账号每秒请求数默认10个，可以升级到100个

意味着：

比如此处的PySpider中的 `rate/burst`



中的表示每秒几个请求的rate，不能超过上面的限制

即：此处rate如果设置了超过买的套餐的次数限制，后续请求会报错 `429 Too many requests`

就像这里的解释：

- 多贝云|动态IP代理拨号代理VPN代理动态HTTP代理
  > 一秒内发起的所有请求数之和，如果超出套餐限制的值，那么多出的部分会被拒绝访问（提示 `429 Too many requests!` ），这时候需要购买更多的请求数以满足业务需求

代理IP检测

另外，对于如何查看到自己当前的IP，可以用：

dobelcloud/PythonRequestsDemo.py at master · dobelgit/dobelcloud

提到的：

https://www.taobao.com/help/getip.php

得到：

```
ipCallback({ip:"222.92.130.218"})
```

或：

隧道代理接入文档 | 蜻蜓代理 - 企业级高质量代理ip平台

提到的：

https://proxy.horocn.com/api/ip

```
222.92.130.218
```

代码测试IP是否相同

在买了动态IP代理后，希望搞清楚是否生效，即每次获取的IP是否的确不同，可以自己写代码测试。

下面是我的测试代码，以阿布云的代理为例，供参考：

```
# test proxy IP

import requests

#目标网址
targetUrl = "http://httpbin.org/get"
# targetUrl = "http://test.abuyun.com"
# targetUrl = "https://www.taobao.com/help/getip.php"
# targetUrl = "http://proxy.abuyun.com/switch-ip"
# targetUrl = "http://proxy.abuyun.com/current-ip"

#http代理接入服务器地址端口
proxyHost = "http-dyn.abuyun.com"
proxyPort = "9020"

#账号密码
proxyUser = "password1"
proxyPass = "password2"

proxyMeta = "http://%(user)s:%(pass)s@%(host)s:%(port)s" % {
  "host" : proxyHost,
  "port" : proxyPort,
  "user" : proxyUser,
  "pass" : proxyPass,
}

proxies = {
  "http"  : proxyMeta,
  "https" : proxyMeta,
}

result = requests.get(targetUrl, proxies proxies)
# result = requests.get(targetUrl)

print("result=%s" % result)
```

```
print("result.status_code=%s" % result.status_code)
print("result.text=%s" % result.text)
```

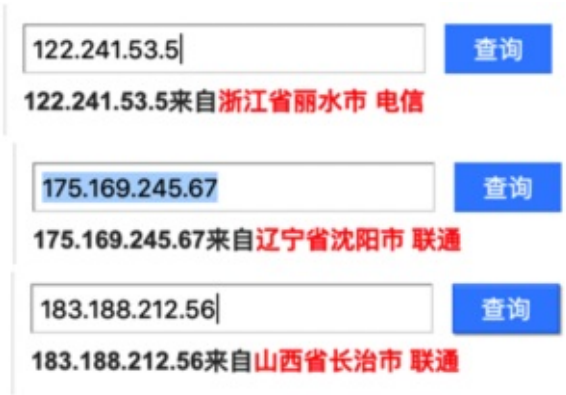
效果是所希望的，正常的，每次IP都不同：

```
result=<Response [200]>
result.status_code=200
result.text={
  "args": {},
  "headers": {
    "Accept": "*/*",
    "Accept-Encoding": "gzip",
    "Host": "httpbin.org",
    "User-Agent": "python-requests/2.19.1"
  },
  "origin": "122.241.53.5, 122.241.53.5",
  "url": "https://httpbin.org/get"
}

result=<Response [200]>
result.status_code=200
result.text={
  "args": {},
  "headers": {
    "Accept": "*/*",
    "Accept-Encoding": "gzip",
    "Host": "httpbin.org",
    "User-Agent": "python-requests/2.19.1"
  },
  "origin": "175.169.245.67, 175.169.245.67",
  "url": "https://httpbin.org/get"
}

result=<Response [200]>
result.status_code=200
result.text={
  "args": {},
  "headers": {
    "Accept": "*/*",
    "Accept-Encoding": "gzip",
    "Host": "httpbin.org",
    "User-Agent": "python-requests/2.19.1"
  },
  "origin": "183.188.212.56, 183.188.212.56",
  "url": "https://httpbin.org/get"
}
```

查查这几个IP都是哪里的：



看起来效果不错。

免费试用

IP代理提供商 注册后 往往都有免费试用代理供测试比如：

- 蘑菇代理 - 购买API代理
  > 您只需要注册我们的平台账号，成功注册后即可申请免费试用
  >
  > 私密API代理提供500个IP，1小时的测试时间;HTTP隧道代理提供10个并发，3小时的测试时间

甚至有些不注册，也提供少数的供测试的代理IP，比如：

- 国内高匿免费HTTP代理IP - 快代理
  > 免费代理由第三方服务器提供，IP不确定性较大，总体质量不高。如需购买基于自营服务器的高质量IP产品，请联系客服开通测试订单

所以在确定买哪家代理之前，调研和对比期间，可以测试其试用代理效果如何，再做决定。

服务商

接着介绍国内常见的具体的代理服务商，即卖代理的公司。

各家代理对比

如果想要大概了解每家代理效果如何，可以参考这里的测评对比。

下面列出对比结果供参考：

代理商家	套餐类型	可用率	可用率评价	响应时间均值	响应速度评价	响应时间方差	稳定性	包月价格	价格评价	安全性	访问频率限制	调取频率限制	推荐指数
芝麻 HTTP 代理	默认版	99%	极高	0.916853	极快	1.331989	极好	360	较高	高	无	1 秒	★★★★★
阿布云代理	专业版	90.4%	高	0.68770707	极快	1.1477163	极好	429	高	高	有	无需获取	★★★☆
	动态版	98.8%	高	1.83994	快	6.0491614	好	429	高	高	有	无需获取	★★★★
	经典版	99.8%	极高	0.49301904	极快	0.25724468	极好	429	高	高	有	无需获取	★★★★
多贝云代理	套餐一	100%	极高	0.658007	极快	0.199466	极好	500	高	高	有	无需获取	★★★★★
	套餐二	100%	极高	0.510748	极快	0.022519	极好	600	高	高	有	无需获取	★★★★★
	套餐三	100%	极高	0.678544	极快	0.519705	极好	425	高	高	有	无需获取	★★★★☆
大象代理	个人版	47.6%	低	5.340489	慢	78.56444	一般	98	低	低	无	1 秒	★★
	专业版	56.8%	一般	6.87078	慢	105.7984	差	198	较低	低	无	1 秒	★☆
	企业版	51.8%	一般	6.3081837	慢	121.08402	差	498	高	低	无	无限制	★
全网代理	普通版	44%	低	5.584057	慢	47.442596	一般	93	低	低	无	无限制	★★
	动态版	97%	高	2.776973	一般	17.568045	一般	160	较低	低	无	100毫秒	★★★
快代理	VIP 套餐	35.6%	一般	16.636587	极慢	221.69661	差	200	中	低	无	200毫秒	☆
蘑菇代理	默认版	99.4%	极高	1.0985725	快	9.532586	好	169	较低	低	无	5秒	★★★★☆
太阳 HTTP 代理	默认版	80%	一般	1.2522483	快	12.662229	一般	198	较低	高	无	1秒	★★★★
讯代理	优质代理	99%	极高	1.0512681	快	6.4247565	好	210	中	低	无	5秒	★★★★☆
	混播代理	98.8%	高	1.0664985	快	6.451699	好	729	高	低	无	10秒	★★★☆
	独享代理	100%	极高	0.7056521	极快	0.35416448	极好	210	中	低	无	15秒	★★★★☆
云代理	VIP 套餐	97.8%	高	3.4216988	一般	38.120296	一般	120	较低	低	无	无限制	★★★☆
站大爷代理	普通代理	18.4%	极低	5.067193	慢	66.12128	一般	80	低	低	无	3秒	★☆
	短效优质代理	97.6%	高	1.5625348	快	8.121197	好	475	高	低	无	10秒	★★★☆
西刺代理	免费	6.2%	极低	9.712833	慢	95.09569	一般	0	免费	低	无	无限制	★

所以在综合来看比较推荐的有：芝麻代理、讯代理、阿布云、多贝云代理，详细的对比结果可以参照表格。

阿布云代理

之前自己为了反扒，去买了阿布云的代理，实现了每次请求IP都不同的动态IP的效果，效果还不错。

- 阿布云代理
  - 主页
    - HTTP隧道产品
      - https://www.abuyun.com/http-proxy/products.html

购买阿布云的HTTP隧道代理

HTTP隧道产品 | 阿布云 - 为大数据赋能

购买 HTTP隧道 动态版

- 服务说明
  - 动态版HTTP隧道服务说明 | 阿布云 - 为大数据赋能
- 接入指南
  - 动态版HTTP隧道接入指南 | 阿布云 - 为大数据赋能
    - Python
      - HTTP隧道（动态版）Python 接入指南| 阿布云 - 为大数据赋能

先去注册，进入后台：

点击了 申请免费测试隧道 后：

即可申请到 免费测试时间：4个小时 的动态IP。

经过测试无误后，即可后续继续购买。

比如 购买1天的：

跳转到支付宝付款，成功付款后返回订单列表：

进入云隧道->HTTP代理，可看到已购产品列表：

得到了 `通行证书` 和 `通行密钥`，就是**用户名**和**密码**，即可放在代码中使用。

账号过期

如果后续所购服务过期了仍继续调用接口，则会报错 `HTTP 402: Payment Required`。

比如 `PySpider` 中的报错：

```
[W 190502 10:04:59 tornado_fetcher:423] [402] DianpingChildrenEnglish:03d7573282446a107bf426f38e30f406 http://www.dianping.com/shop/98532606#221048_745240 0.06s
[E 190502 10:04:59 processor:202] process DianpingChildrenEnglish:03d7573282446a107bf426f38e30f406 http://www.dianping.com/shop/98532606#221048_745240 -> [402] len:312 -> result:None fol:0 msg:0 err:HTTPError('HTTP 402: Payment Required',)
```

解决办法：重新续费，或者购买新服务，再去代码中换上新的 `通行证书` 和 `通行密钥`，即可。

多贝云代理

之前也去用过

多贝云的HTTP代理

觉得也还可以。

下面记录购买和使用过程。

不同套餐对比

- 多贝云
  - 套餐2：每一段时间内的请求，IP都是固定不变的
    - 注：
      - 套餐2的时间是7~10分钟
      - 套餐1是1~2分钟
    - 有效期过了后，会自动更换新IP的
      - 无需手动更换
      - 当然如果想要手动，主动，更换IP，也是支持的
        - 详见：多贝云|动态IP代理拨号代理VPN代理动态HTTP代理
          - 主动切换IP：
            - 整个过程无需变更代理服务器和端口信息，而是通过调用API实现自动切换IP。应用程序通过HTTP隧道请求 http://ip.dobel.cn/switch-ip ，请求执行成功会返回一个新的IP地址及其剩余可用时长，后续的请求将都会通过新的出口IP地址进行转发
          - 查看当前IP
            - 请求 http://ip.dobel.cn/current-ip 能够获取到当前分配到出口IP以及剩余的可用时长信息。
            - 举例
              - `curl -x http-proxy-t2.dobel.cn:9180 -U ProxyUsername:ProxyPassword http://ip.dobel.cn/current-ip`
              - 返回：
              - `{"wanip":"117.69.50.223","resttime":-215}%`
  - 套餐3：每次请求，IP都不同
    - 才是我们希望的，动态IP

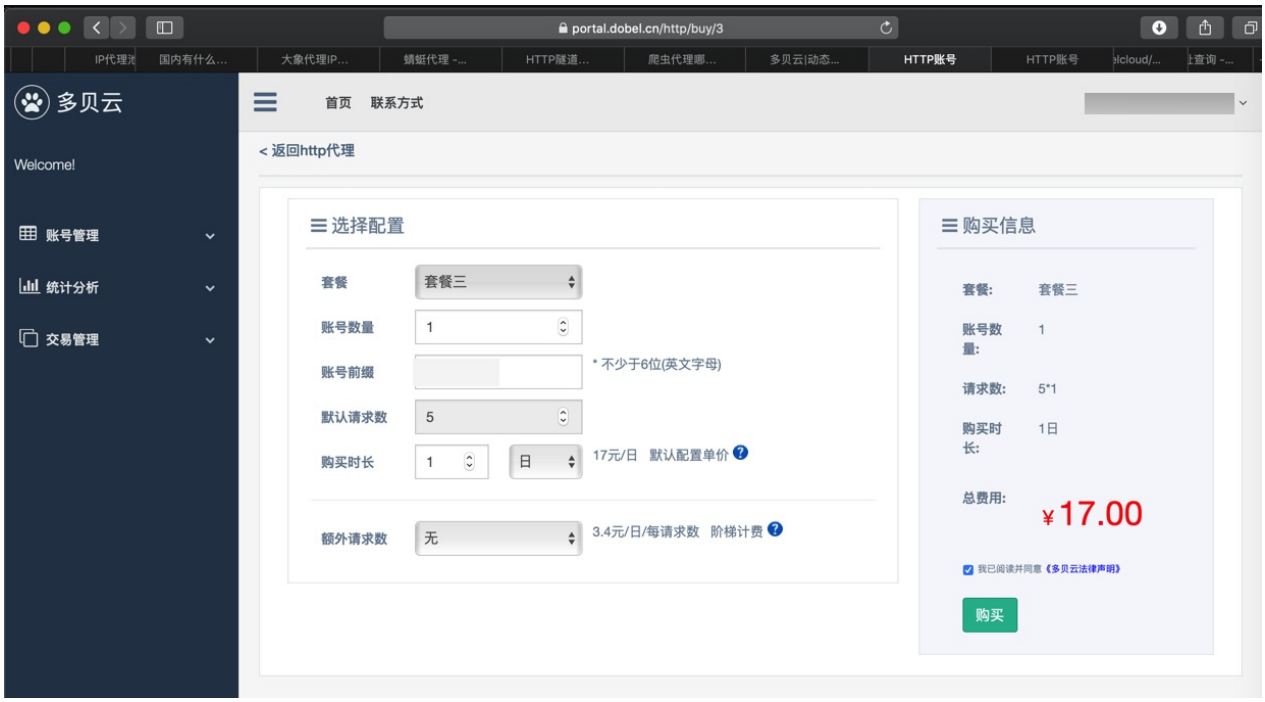
购买多贝云的HTTP代理

从

多贝云|动态IP代理拨号代理VPN代理动态HTTP代理

点击 套餐3

再去选择具体配置：



用支付宝支付后，返回订单：

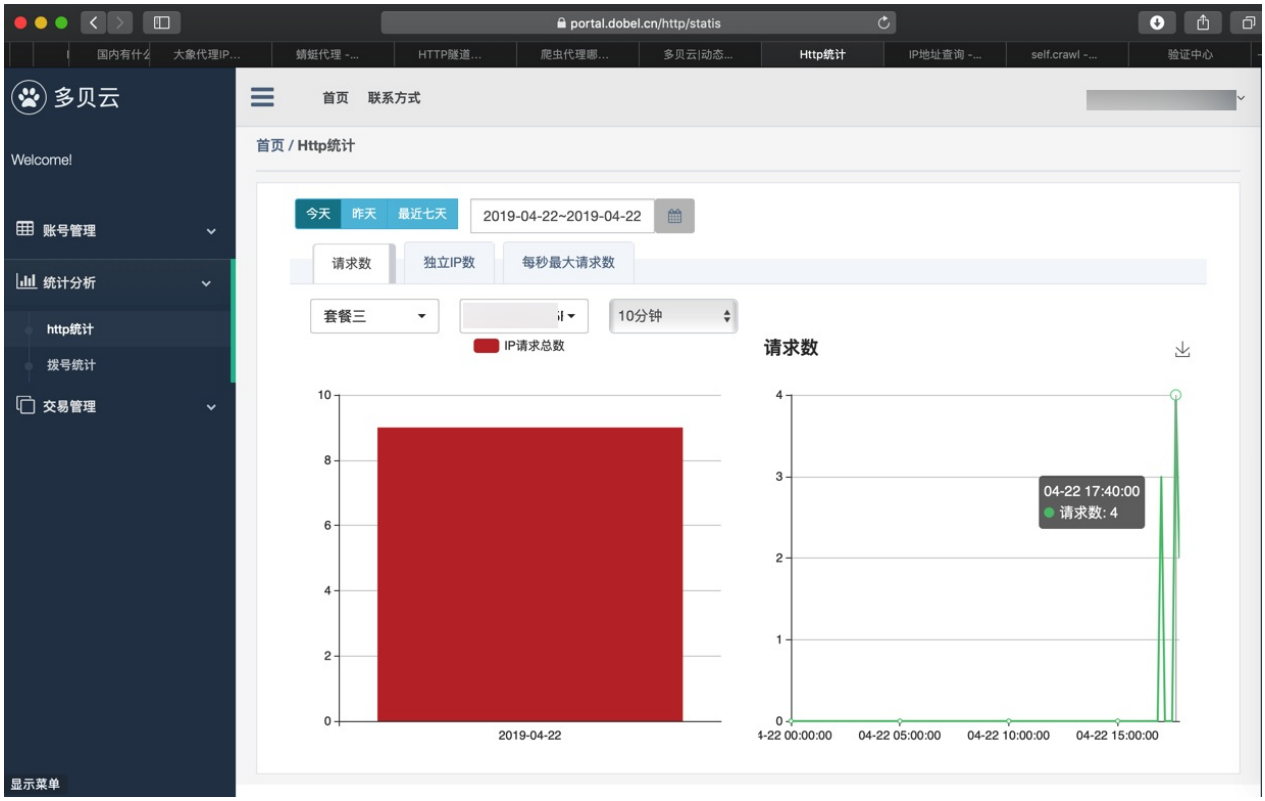
返回 账号管理->http代理，即可看到已购产品：

把其中的**账号**和**密码**放在代码中，即可使用。

使用多贝云

关于如何在代码中使用多贝云代理，可以参考后续章节：PySpider中使用代理

用了一段时间后，还可以去看统计信息：



账号过期

如果账号过期了：

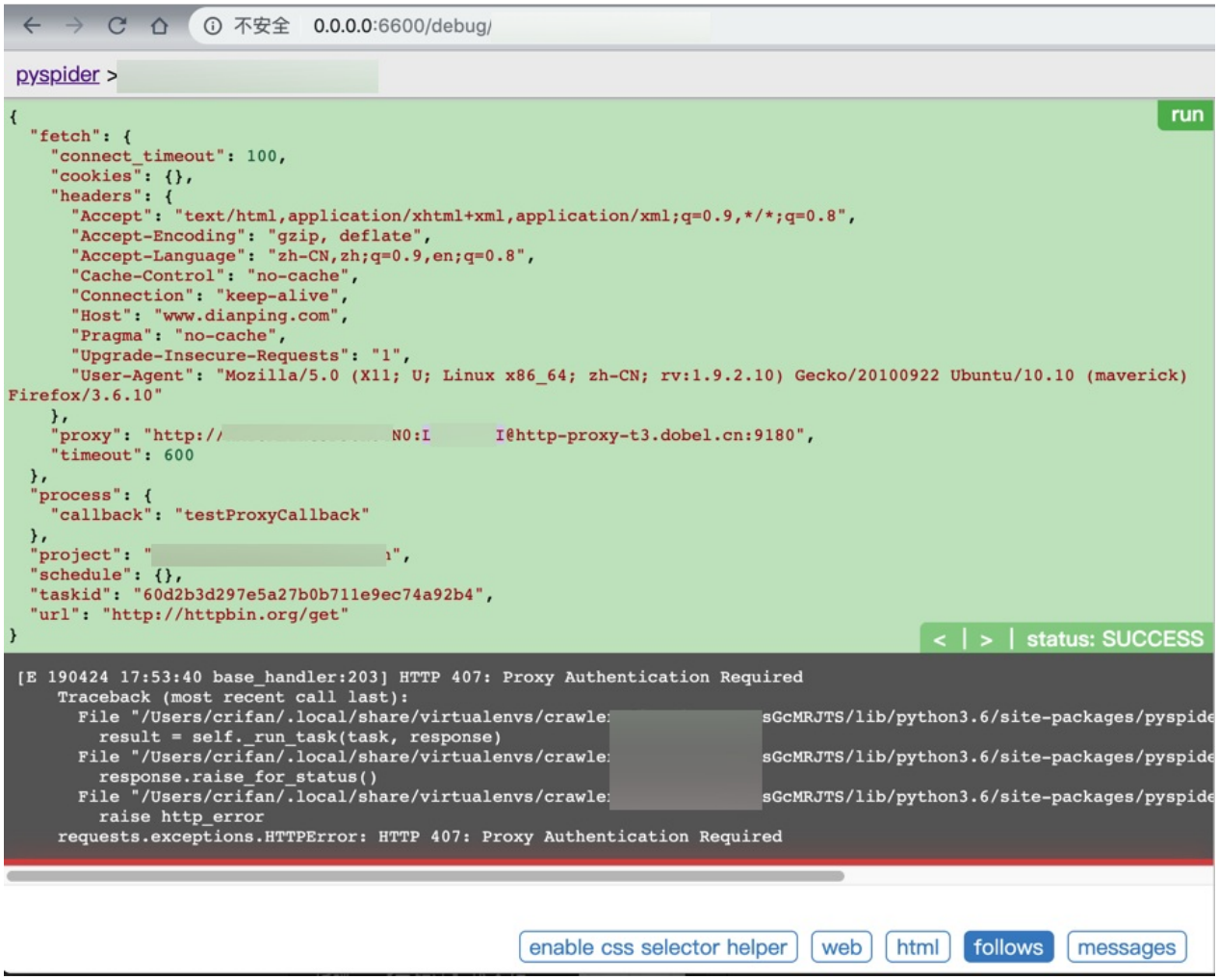
则（PySpider中）会出现：

`requests.exceptions.HTTPError: HTTP 407: Proxy Authentication Required`

的错误：

```
[E 190424 17:53:40 base_handler:203] HTTP 407: Proxy Authentication Required
    Traceback (most recent call last):
      File "/Users/crifan/.local/share/virtualenvs/ crawler_xxx-sGcMRJTS/lib/python3.6/site-packages/pyspider/libs/base_handler.py", line 196, in run_task
        result = self._run_task(task, response)
      File "/Users/crifan/.local/share/virtualenvs/ crawler_xxx-sGcMRJTS/lib/python3.6/site-packages/pyspider/libs/base_handler.py", line 175, in _run_task
        response.raise_for_status()
      File "/Users/crifan/.local/share/virtualenvs/ crawler_xxx-sGcMRJTS/lib/python3.6/site-packages/pyspider/libs/response.py", line 184, in raise_for_status
        raise http_error
```

```
requests.exceptions.HTTPError: HTTP 407: Proxy Authentication Required
```



验证代理是否生效

此处介绍，购买了代理后，如何验证（动态IP）代理是否生效。

其实等价于：IP代理中的认证方式

常见的有几种。此处介绍最简单最基本的：

HTTP基本认证

另外，也从：

- 隧道代理接入文档 | 蜻蜓代理 - 企业级高质量代理ip平台
  > 目前，我们使用的是 HTTP基本认证，在发送请求中，添加 `Proxy-Authorization` 头部，值为： `Basic b64encode("username:password")` 。

得知了，此处一般IP代理提供商

- 蘑菇代理 - 购买API代理
- 动态版HTTP隧道接入指南 | 阿布云 - 为大数据赋能
  - 示例代码都是一样的
    - abuyun/proxy-demo: abuyun cloud proxy demo
    - HTTP隧道（动态版）Python 接入指南| 阿布云 - 为大数据赋能

的认证方式，都是用的是：

`Http Basic Authentication` = `Http基本认证`

-》最通用，也相对最简单的方式

-》所以其他很多http方面的库，比如Python的requests，也才会（内置就）支持

所以只需要传递参数，无需手动自己算base64编码等过程了。

示例代码

多贝云

https://github.com/dobelgit/dobelcloud/blob/master/Python/PythonRequestsDemo.py

```
#! -*- encoding:utf-8 -*-

import requests

#目标网址
targetUrl = "https://www.taobao.com/help/getip.php"

#http代理接入服务器地址端口
proxyHost = "域名"
proxyPort = "端口"

#账号密码
proxyUser = "账号"
proxyPass = "密码"

proxyMeta = "http://%(user)s:%(pass)s@%(host)s:%(port)s" % {
  "host" : proxyHost,
  "port" : proxyPort,
  "user" : proxyUser,
  "pass" : proxyPass,
}
```

```
proxies = {
  "http"  : proxyMeta,
  "https" : proxyMeta,
}

result = requests.get(targetUrl, proxies=proxies)

print result.status_code
print result.text
```

阿布云

https://github.com/abuyun/proxy-demo/blob/master/http-tunnel/http-dyn/python/python3/urllib/proxy-demo.py

```
#! -*- encoding:utf-8 -*-

import requests

# 要访问的目标页面
targetUrl = "http://test.abuyun.com"
#targetUrl = "http://proxy.abuyun.com/switch-ip"
#targetUrl = "http://proxy.abuyun.com/current-ip"

# 代理服务器
proxyHost = "http-dyn.abuyun.com"
proxyPort = "9020"

# 代理隧道验证信息
proxyUser = "H01234567890123D"
proxyPass = "0123456789012345"

proxyMeta = "http://%(user)s:%(pass)s@%(host)s:%(port)s" % {
    "host" : proxyHost,
    "port" : proxyPort,
    "user" : proxyUser,
    "pass" : proxyPass,
}

proxies = {
    "http"  : proxyMeta,
    "https" : proxyMeta,
}

resp = requests.get(targetUrl, proxies=proxies)
print resp.status_code
print resp.text
```

抓包代理

app抓包类的代理，往往要用到抓包工具，比如Charles、mitmproxy，wireshark等。

安装根证书

在给抓包的工具初始化代理环境时，往往会涉及到安装对应工具的根证书，下面总结其中相对通用的逻辑。

- 安装根证书
  - 安装代理证书
    - 给Charles安装根证书
    - 给mitmproxy安装根证书
  - 通用逻辑
    - pc端
      - 开启代理
    - 移动端
      - 给WiFi加上代理
      - 用浏览器访问网址下载代理证书
        - 访问网址
          - Charles地址
            - http://chls.pro/ssl
          - mitmproxy地址
            - http://mitm.it
              - 如果没给WiFi加代理则会提示
                - `if you can see this, traffic is not passing through mitmproxy`
        - 证书文件
          - 正常时
            - Charles: `charles-proxy-ssl-proxying-certificate.pem`
              - 有时是: `getssl.crt`
            - mitmproxy: `mitmproxy-ca-cert.pem`
          - 异常时：无法下载
            - Charles: `downloadfile.crt`
            - mitmproxy: `perm.crt`

科学上网代理

总体逻辑是：

- 先要有个代理服务器
  - 方式
    - 自己搭建
    - 购买网上别人（公司）的（专业服务）
- 再去用客户端使用代理

关于科学上网的细节，详见独立教程：

科学上网相关知识总结

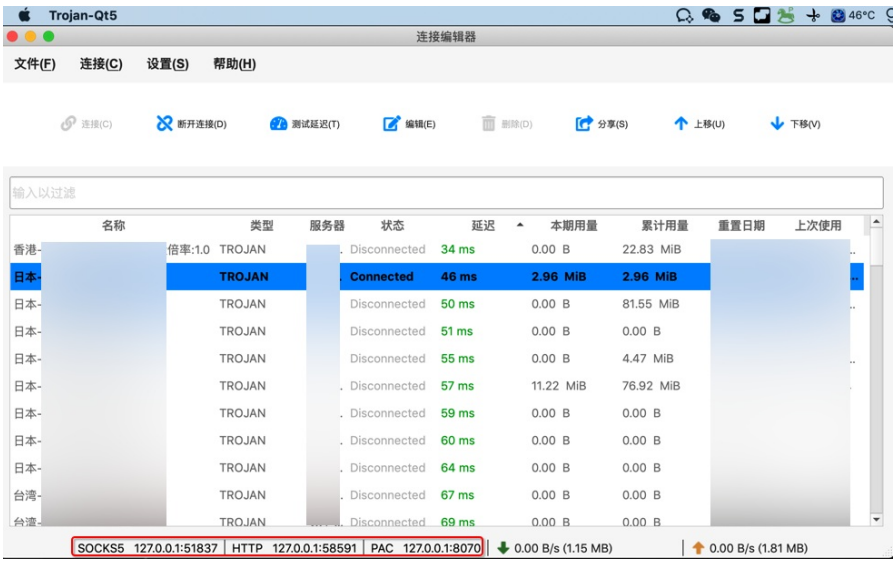
下面介绍如何从科学上网的客户端，获取当前的代理配置信息。

从客户端获取代理配置

此处介绍，从科学上网类的客户端，获取当前代理配置信息。

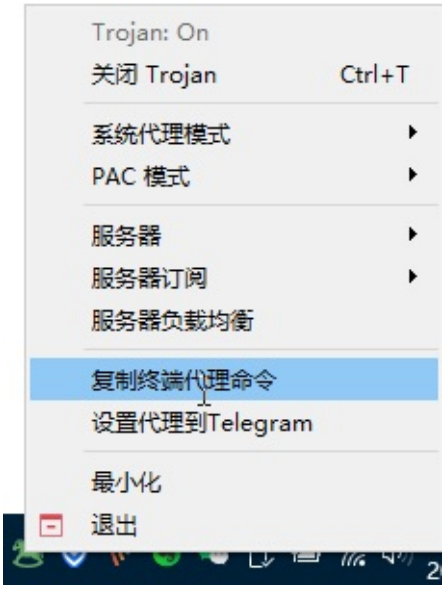
- 科学上网工具的代理信息
  - Shadowsocks类
    - http
      - http://127.0.0.1:1086
    - https
      - https://127.0.0.1:1086
    - socks5
      - socks5://127.0.0.1:1086
    - PAC自动代理
      - http://127.0.0.1:8070/proxy.pac
  - Trojan类
    - `Trojan-QT5`：`Mac` 的 `Trojan-QT5` 、 `Windows` 的 `trojan-qt5.exe`
      - 得到代理配置的方式
        - 从窗口底部看到
          - 步骤
            - 打开窗口
            - 

            - 窗口底部有显示当前代理配置信息



- 看到的的结果
  - SOCK5 127.0.0.1:51837
  - HTTP 127.0.0.1:58591
  - PAC 127.0.0.1:8070
- => 意味着完整代理地址是
  - SOCK5
    - socks5://127.0.0.1:51837
  - HTTP
    - http://127.0.0.1:58591
    - https://127.0.0.1:58591
  - PAC = 自动模式 = 自动代理
    - http://127.0.0.1:8070/proxy.pac

- 从菜单中拷贝得到
  - 步骤
    - 托盘图标 -> `复制终端代理命令`
      - Win



      - Mac

- 

  - 新版同时支持 `Mac` 和 `Win` 的配置
  - 

  - 拷贝出的值是
    - `Mac`

```
export HTTP_PROXY http://127.0.0.1:58591; export HTTPS_PROXY http://127.0.0.1:58591;
export ALL_PROXY socks5://127.0.0.1:51837
```

    - `Windows`

```
export HTTP_PROXY http://127.0.0.1:1081; export HTTPS_PROXY http://127.0.0.1:1081;
export ALL_PROXY socks5://127.0.0.1:1080
```

  - 如何用
    - 具体使用方式，详见后续章节：如何添加代理

如何添加代理

编程语言

Python

requests

python的网络库requests中加proxy代理

前提：确保自己的代理正常。比如此处的 `V2RayU` 中的代理：

代码：

此处给（调用WordPress的REST的api时的）网络请求加上代理：

```
cfgProxies = {
    "http"  : "http://127.0.0.1:1087",
    "https" : "http://127.0.0.1:1087",
}

resp = requests.post(
    createPostUrl,
    proxies=cfgProxies,
    headers=curHeaders,
    json=postDict,
)
```

即可。

PySpider中使用代理

爬虫框架PySpider内部用的是 `requests` ，其设置代理的方式，也就是和requests一样了。

PySpider中设置了全局的代理：

```
### dobel 多贝云 IP代理
#http代理接入服务器地址端口
ProxyHost = "http-proxy-t3.dobel.cn"
ProxyPort = "9180"
```

```
#账号密码
ProxyUser = "YourUserName"
ProxyPass = "YourPassword"

ProxyUri = "http://%(user)s:%(pass)s@%(host)s:%(port)s" % {
        "host" : ProxyHost,
        "port" : ProxyPort,
        "user" : ProxyUser,
        "pass" : ProxyPass,
}

class Handler(BaseHandler):
    crawl_config = {
        "proxy": ProxyUri,
        '''
    }
```

之后的 `self.crawl` 正常调用url，即可用上此处的代理了。

其中 `ProxyUri` 是这种：

```
http://YourUserName:YourPassword@http-proxy-t3.dobel.cn:9180
```

开发工具

git

git 添加 代理 加速

- 简述：

```
git config --global http.proxy socks5://127.0.0.1:1086
git config --global --unset http.proxy
```

- 详解：
  - git中没有可以设置，当前这次的，临时的，只有一次的代理
  - 只能间接的实现，此处单独针对于GitHub的
    - 全局的设置代理：仅针对 https://github.com 使用代理

      ```
      git config --global http.https://github.com.proxy socks5://127.0.0.1:1086
      ```

    - 用 `git clone` 下载时，就可以用上代理实现下载加速了

      ```
      git clone https://github.com/appium/WebDriverAgent.git
      ```

    - 用完再去取消代理

      ```
      git config --global --unset http.https://github.com.proxy
      ```

- 说明
  - 代理协议
    - 可以把上述的 `socks5://127.0.0.1:1086` 换成其他协议的代理，比如 `http` 的 `http://127.0.0.1:1086`
  - 代理端口
    - 上述端口是：`1086`
      - 如果你的端口不是 `1086`，千万记得要更换成自己的代理的端口
  - 查看当前git的代理配置
    - 随时可以用

      ```
      cat ~/.gitconfig
      ```

    - 查看当前全局的设置，是否正确
      - 比如前面，只给github加上对应代理后，效果是

        ```
        ➜  ~ cat ~/.gitconfig
        # This is Git's per-user configuration file.
        [user]
        # Please adapt and uncomment the following lines:
          name = crifanli
          email = admin@crifan.com
        [http "https://github.com"]
          proxy = socks5://127.0.0.1:51837
        [core]
          autocrlf = input
        ```

    - 如果后续取消了对应代理，则就看不到对应字段了

      ```
      [http "https://github.com"]
        proxy = socks5://127.0.0.1:1086
      ```

给git加代理使用心得

error: RPC failed; curl 35 LibreSSL SSL_connect

如果git下载报错

`error: RPC failed; curl 35 LibreSSL SSL_connect`

可以多试几次，或许就好了。

比如，之前遇到过几次，git下载报错了：

```
➜  temp git clone https://github.com/fcamblor/oh-my-zsh-agnoster-fcamblor.git
Cloning into 'oh-my-zsh-agnoster-fcamblor'...
error: RPC failed; curl 35 LibreSSL SSL_connect: SSL_ERROR_SYSCALL in connection to github.com:443
fatal: the remote end hung up unexpectedly
➜  temp git clone https://github.com/fcamblor/oh-my-zsh-agnoster-fcamblor.git
Cloning into 'oh-my-zsh-agnoster-fcamblor'...
remote: Enumerating objects: 136, done.
remote: Total 136 (delta 0), reused 0 (delta 0), pack-reused 136
Receiving objects: 100% (136/136), 1.14 MiB | 741.00 KiB/s, done.
Resolving deltas: 100% (59/59), done.
```

此时并没有改动代理服务器节点，而是多试了一次（或多次），就又好了。

只对某些url用代理，不对另外一些url用代理

比如：只对 `github` 用代理，而对 `gitee` 不用代理

修改配置为：

```
[http]
    proxy = socks5://127.0.0.1:1086
[http "https://gitee.com/"]
    proxy =
```

pip

如果发现pip下载很慢，可以考虑加代理，提高下载速度：

给pip加单次的临时的代理，以提高下载速度

给pip临时的、单次的加代理，可以用 `--proxy proxy_address`

单次加代理：

```
pip --proxy http://127.0.0.1:1087 install pandas
```

用法举例：

```
pip --proxy http://127.0.0.1:1087 install requests pyyaml uiautomator2 selenium tldextract pymongo redis pandas numpy

pip --proxy http://127.0.0.1:1087 install --upgrade pandas==0.24.2

pip --proxy http://127.0.0.1:1087 install pocoui
```

Mac/Windows中给pip添加全局代理，以提高下载速度

从Trojan客户端获取到代理命令后，去运行：

- Windows

  ```
  set HTTP_PROXY http://127.0.0.1:1081
  set HTTPS_PROXY http://127.0.0.1:1081
  set ALL_PROXY socks5://127.0.0.1:1080
  ```

- Mac/Linux

  ```
  export HTTP_PROXY http://127.0.0.1:58591
  export HTTPS_PROXY http://127.0.0.1:58591
  export ALL_PROXY socks5://127.0.0.1:51837
  ```

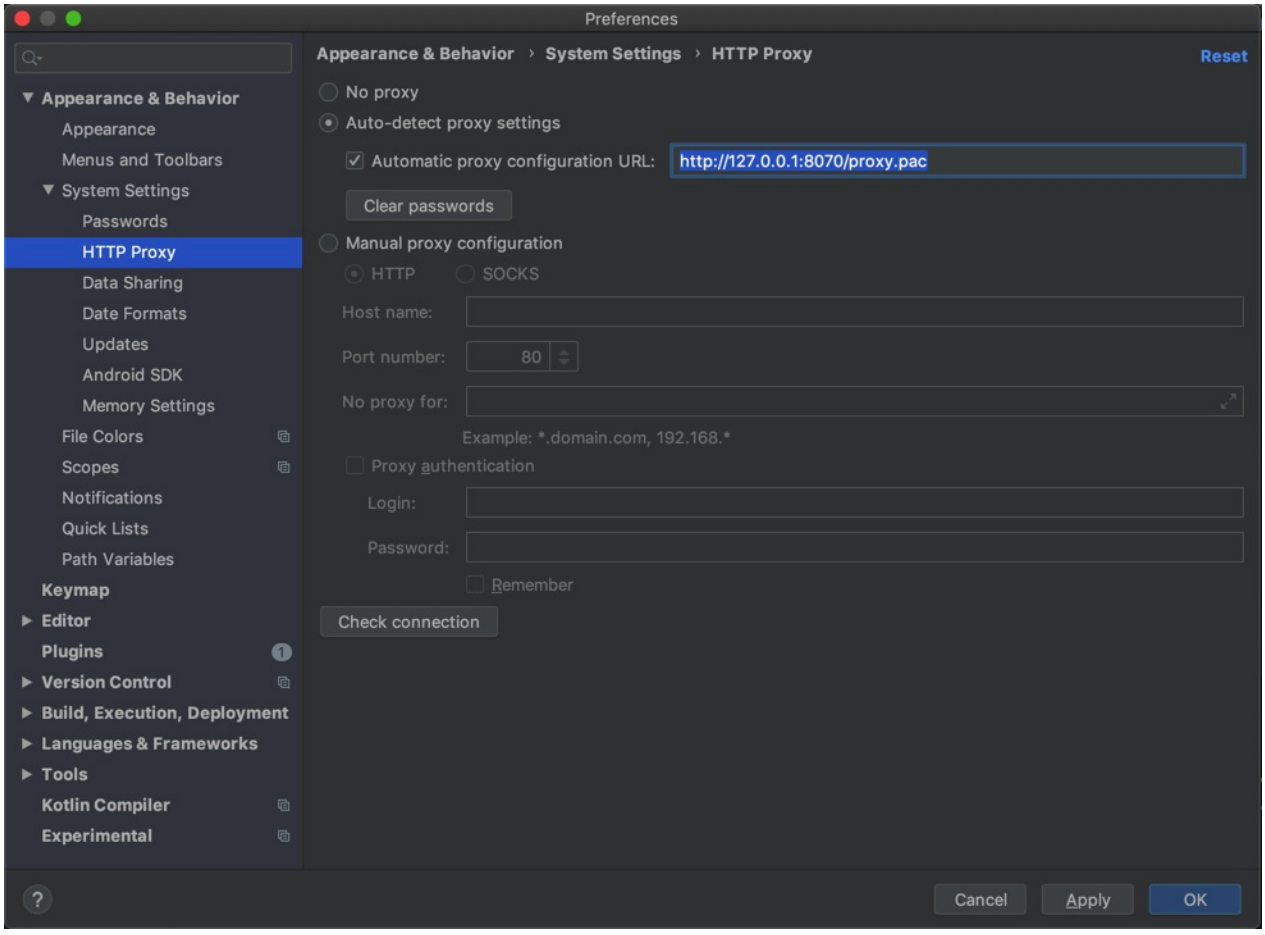
详见：桌面端 · 网络中转站：代理技术

Android Studio

给android studio加代理

设置为 **PAC自动代理**：

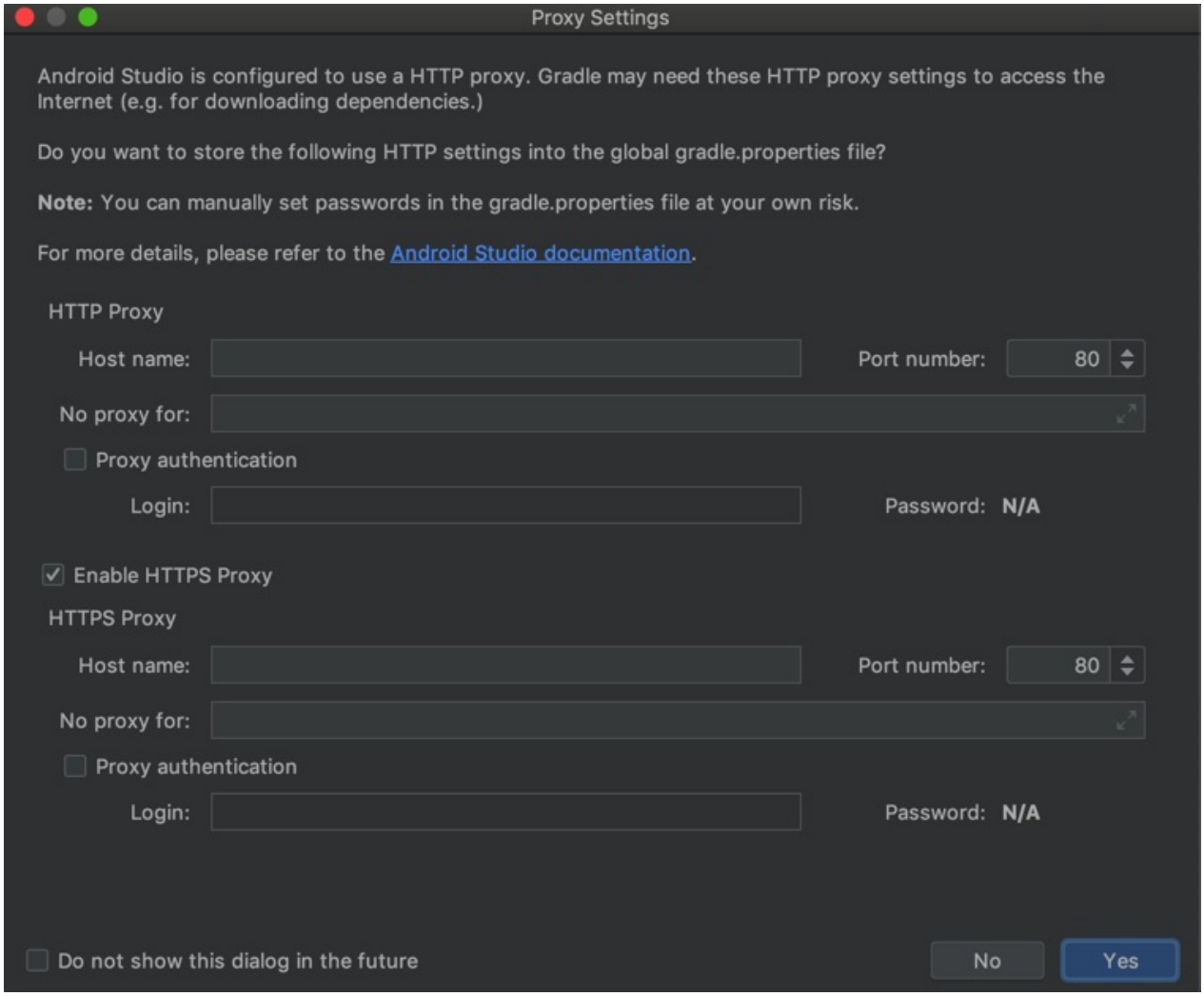
- `Appearance & Behavior` -> `System Settings` -> `HTTP Proxy` -> `Auto-detect proxy settings`：
  - 勾选☑：`Automatic proxy configuration URL`
    - 值为：`http://127.0.0.1:8070/proxy.pac`



gradle

android studio中给gradle加代理

升级了Android Studio，弹框问是否要设置Gradle的代理：



其中的：

- HTTP
  - host
  - No proxy for
  - Proxy authentication
    - Login
    - Password
- HTTPS
  - host
  - No proxy for
  - Proxy authentication
    - Login
    - Password

等内容，其实就是对应着：

- gradle官网

- Accessing the web through a HTTP proxy

中提到的：

- Configuring an HTTP proxy using gradle.properties

```
systemProp.http.proxyHost www.somehost.org
systemProp.http.proxyPort 8080
systemProp.http.proxyUser userid
systemProp.http.proxyPassword password
systemProp.http.nonProxyHosts *.nonproxyrepos.com localhost
```

和：

- Configuring an HTTPS proxy using gradle.properties

```
systemProp.https.proxyHost www.somehost.org
systemProp.https.proxyPort 8080
systemProp.https.proxyUser userid
systemProp.https.proxyPassword password
systemProp.https.nonProxyHosts *.nonproxyrepos.com localhost
```

的 系统属性 `systemProp` 中的 `http` 和 `https` 的

- proxyHost
- proxyPort

如果要登录认证，再需要：

- proxyUser
- proxyPassword

以及有哪些特殊的url需要被排除掉，可以加到：

- nonProxyHosts

而此处，想要全局的gradle都用上代理，可以去把这部分代理配置，加到：

- gradle的全局配置文件
  - 当前用户下面的 `gradle.properties` 中
    - Linux系（Linux/Mac等）
      - `$HOME/.gradle/gradle.properties`
    - Win系
      - `%userprofile%\.gradle\gradle.properties`

在其中加上对应配置，比如：

```
systemProp.http.proxyHost 127.0.0.1
systemProp.http.proxyPort 58591
systemProp.https.proxyHost 127.0.0.1
systemProp.https.proxyPort 58591
```

> 注：
>
> 其中此处的proxyHost和proxyPort，是我此处 `Trojan` 的代理配置。
>
> （可以在 `Trojan-QT5` 的界面中看到，或从菜单 `复制终端代理命令` 中拷贝出来）

rsync

给rsync加代理加速，即可以通过给rsync加代理，实现加快文件同步上传的速度。

具体方式是：

- 加代理之前
  - sshpass -f /Users/crifan/dev/dev_root/gitbook/gitbook_src_root/common/config/deploy/deploy_server_password.txt rsync -avzh --progress --stats --delete --force /Users/crifan/dev/dev_root/gitbook/gitbook_src_root/generated/books/android_app_security_crack/release/android_app_security_crack root@149.28.136.189:/data/wwwroot/book.crifan.com/books
- 加代理之后
  - sshpass -f /Users/crifan/dev/dev_root/gitbook/gitbook_src_root/common/config/deploy/deploy_server_password.txt rsync -avzh --progress --stats --delete --force **-e "ssh -o 'ProxyCommand nc -X 5 -x 127.0.0.1:51837 %h %p' -o ServerAliveInterval=30 -o ServerAliveCountMax=5"** /Users/crifan/dev/dev_root/gitbook/gitbook_src_root/generated/books/android_app_security_crack/release/android_app_security_crack root@149.28.136.189:/data/wwwroot/book.crifan.com/books

其中参数含义解释：

- rsync
  - `-e, --rsh=COMMAND`
    - specify the remote shell to use
      - `ssh -o 'ProxyCommand nc -X 5 -x 127.0.0.1:51837 %h %p' -o ServerAliveInterval=30 -o ServerAliveCountMax=5`
- ssh
  - `-o option`
    - Can be used to give options in the format used in the configuration file. This is useful for specifying options for which there is no separate command-line flag. For full details of the options listed below, and their possible values
      - ProxyCommand
- `nc -X 5 -x 127.0.0.1:51837 %h %p`
  - 参数语法
    - `-X proxy_version`
      - Requests that nc should use the specified protocol when talking to the proxy server. Supported protocols are "4" (SOCKS v.4), "5" (SOCKS v.5) and "connect" (HTTPS proxy). If the protocol is not specified, SOCKS version 5 is used.
    - `-x proxy_address[:port]`
      - Requests that nc should connect to hostname using a proxy at proxy_address and port. If port is not specified, the well-known port for the proxy protocol is used (1080 for SOCKS, 3128 for HTTPS).
  - 参数含义
    - `-X 5`
      - SOCKS 5版协议
        - 此处用的是SOCKS5代理（不是http代理）
    - `-x 127.0.0.1:1080`
      - 代理地址和端口是：127.0.0.1:1080
    - `%h %p`
      - 应该是对应着：`[hostname] [port[s]]`
        - 分别表示：
          - `%host`：当前主机 host？
          - `%p`：当前端口 port？

- ssh_config
  - 参数语法
    - `ServerAliveInterval`
      - Sets a timeout interval in seconds after which if no data has been received from the server, ssh(1) will send a message through the encrypted channel to request a response from the server. The default is 0, indicating that these messages will not be sent to the server.
    - `ServerAliveCountMax`
      - Sets the number of server alive messages (see below) which may be sent without ssh(1) receiving any messages back from the server. If this threshold is reached while server alive messages are being sent, ssh will disconnect from the server, terminating the session. It is important to note that the use of server alive messages is very different from TCPKeepAlive(below). The server alive messages are sent through the encrypted channel and therefore will not be spoofable. The TCP keepalive option enabled by TCPKeepAlive is spoofable. The server alive mechanism is valuable when the client or server depend on knowing when a connection has become unresponsive.
      - The default value is 3. If, for example, ServerAliveInterval (see below) is set to 15 and ServerAliveCountMax is left at the default, if the server becomes unresponsive, ssh will disconnect after approximately 45 seconds.
  - 参数含义
    - `ServerAliveInterval=30`
      - 每次最多30秒
    - `ServerAliveCountMax=5`
      - 最多5次

SecureCRT

给SecureCRT的ssh加代理

背景：用SecureCRT的ssh登录操作远程的CentOS的VPS服务器

但是速度太慢了，现象是：操作太卡了，输入命令反应太慢了。

希望能给SecureCRT的ssh加上（科学上网的）代理，提高访问速度。

实现方式： 通过Firewall实现ssh的proxy代理

具体步骤：

此处需要去给 `全局配置` = `Global Options`

加上配置。

点击Add

弹出对话框： `FirewallPropertiesDialog`

点击 `Type` 的 `Generic/Telnet Proxy` ，出现各种选项，此处选择 `SOCKS5 version 5(no authentification)` ：

表示是用的 `SOCKS` 的协议，且无用户名密码的验证。

此处用前面的从客户端获取代理配置 · 网络中转站：代理技术获取到了代理配置，比如：

```
export HTTP_PROXY http://127.0.0.1:58591; export HTTPS_PROXY http://127.0.0.1:58591; export ALL_PROXY socks5://127.0.0.1:51837
```

此处去填上对应代理配置信息：

- `Name` ：随便起个名字
  - 比如： `local_socks5`
- `Type` ： `SOCKS version 5 (no authentication)`
- `Parameters`
  - `Hostname or IP` ： `127.0.0.1`
  - `Port` ： `51837`

再点击 `OK` ，即可返回代理配置列表：

再编辑当前的 `Session Options`

此处Firewall是灰色，无法修改

所以需要先去断开连接，再去修改

断开后，就可以去选择刚新建的Firewall：`local_socks5`

即可成功设置防火墙=此处的代理：

然后再去输入命令就会很顺畅，不会卡顿了：

移动端

给移动端设置代理之前，往往要：

- 主机
  - 确保已开代理服务
    - 比如
      - Mac中运行了mitmdump代理服务
      - Mac中打开了Charles软件
  - 知道当前主机的IP地址

查看当前主机的IP

在给移动端（手机等）设置代理之前，往往对应的代理IP地址都是PC端主机地址。

所以需要去查看当前的主机的IP地址。

Mac中查看自己的IP地址的方式可以用命令行：

```
crifanli@crifanlideMac ~ ifconfig | grep 192.168
    inet 192.168.17.128 netmask 0xffffff00 broadcast 192.168.17.255
```

其中可以看出，Mac中的WiFi网络的IP是：

`192.168.17.128`

Android

此处介绍如何给Android手机中的WiFi去设置代理。

核心逻辑是：进入WiFi详情页，开启代理，输入代理配置信息，即可。

比如：

- 华为畅享6S
  -
- 锤子M1L
  - 步骤：设置->无线网络->点击你要设置代理的Wifi->高级设置
  - WiFi代理配置信息
    - 代理：`手动`
    - 设置Charles中的代理配置：
      - 代理服务器主机名：`10.108.129.57`
      - 代理服务器端口：`5678`
  - 图

iOS

下面介绍，如何在iOS的iPhone中，设置代理。

iPhone中给WiFi设置添加代理

从设置中：

点击对应的WiFi，进入无线局域网：

点击当前的WiFi，进入Wifi详情页。

默认是 `HTTP代理->配置代理：关闭`

点击配置代理，进入设置页：

点击 手动 ，出现 服务器 、 端口 、 鉴定 等选项：

然后输入对应的代理配置信息：

再点击右上角的 `存储`，即可保存并返回上一页，看到 `配置代理` 已变成 `手动`：

即表示已手动设置好代理了。

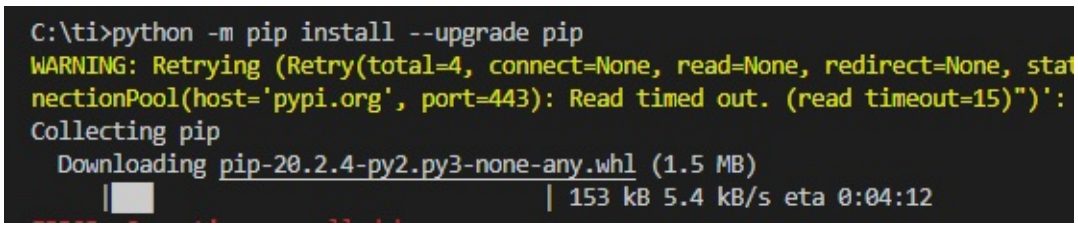
桌面端

Mac/Windows等系统中如何使用科学上网类工具的代理

背景

win10中用python的pip去安装库，但是速度太慢：

```
C:\ti python -m pip install --upgrade pip
...
Collecting pip
  Downloading pip-20.2.4-py2.py3-none-any.whl (1.5 MB)
     |                                | 153 kB 5.4 kB/s eta 0:04:12
```



可以考虑给系统全局加代理，以提高pip的下载速度。

当然，也会给其他通过环境变量检测是否有代理的工具，用上对应的代理。

如何添加全局代理

在前序章节从客户端获取代理配置 · 网络中转站：代理技术获取到代理配置后，下面介绍如何使用代理

- 总体逻辑是：设置对应的 `HTTP_PROXY` 、 `HTTPS_PROXY` 、 `ALL_PROXY` 环境变量
- 具体方式
  - `Mac` / `Linux`
    - 去终端 `Terminal` 中，运行上述命令
      - 单行一次性运行

        ```
        export HTTP_PROXY http://127.0.0.1:58591; export HTTPS_PROXY http://127.0.0.1:58591; export ALL_PRO
        XY socks5://127.0.0.1:51837
        ```

      - 或 多行分别运行

        ```
        export HTTP_PROXY http://127.0.0.1:58591
        export HTTPS_PROXY http://127.0.0.1:58591
        export ALL_PROXY socks5://127.0.0.1:51837
        ```

  - `Windows`
    - 需要先把（Linux类的系统中的） `export` 改为（Windows中设置环境变量的） `set` ，再去运行对应命令

      ```
      set HTTP_PROXY http://127.0.0.1:1081
      set HTTPS_PROXY http://127.0.0.1:1081
      set ALL_PROXY socks5://127.0.0.1:1080
      ```

    - 注
      - Windows中如何确认变量已正确设置，可以用

        ```
        set HTTP_PROXY
        set HTTPS_PROXY
        set ALL_PROXY
        ```

    - Windows中取消代理
        - `set` 为空值

```
set HTTP_PROXY=
set HTTPS_PROXY=
set ALL_PROXY=
```

        - 或 `unset`

```
unset HTTP_PROXY
unset HTTPS_PROXY
unset ALL_PROXY
```

- 内部逻辑和含义
    - `HTTP_PROXY`：所有的 `http` 的请求都用此代理 `http://127.0.0.1:58591`
    - `HTTPS_PROXY`：所有的 `https` 的请求都用此代理 `http://127.0.0.1:58591`
    - `ALL_PROXY`：所有的、任何的网络请求都通过此代理 `socks5://127.0.0.1:51837`

加代理后的效果

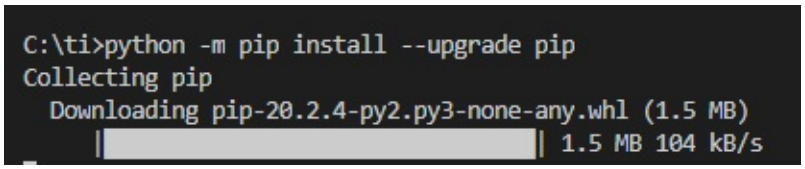
后续Python的pip

`python -m pip install --upgrade pip`

即可自动用上代理，实现下载加速。

此处效果很明显，从默认的pypi的官网下载库文件的速度，从之前的 `5.4 kB/s` 变成现在的 `104 kB/s`，很快就下载完毕了：

```
C:\ti python -m pip install --upgrade pip
Collecting pip
  Downloading pip-20.2.4-py2.py3-none-any.whl (1.5 MB)
     |████████████████████████████████| 1.5 MB 104 kB/s
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 20.1.1
    Uninstalling pip-20.1.1:
      Successfully uninstalled pip-20.1.1
Successfully installed pip-20.2.4
```



Mac

Mac中的WiFi网络和浏览器自动代理

在用Trojan等客户端，实现科学上网后

- Mac中的WiFi网络，自动加上了代理配置：
  - 自动PAC代理
    - 

      - 自动代理配置
        - http://127.0.0.1:8070/proxy.pac

然后Mac中，后续用Safari/Chrome浏览器上网，即可自动用上代理，实现科学上网了。

brew

Mac中想要临时用代理给brew加速，则可以在终端设置全局变量 `ALL_PROXY`：

```
export ALL_PROXY socks5://127.0.0.1:1086
```

后续使用 `brew` 时，即可自动用上代理，实现加速下载了。

说明：

- 此处用的是 `SSR` 的 `SOCKS5` 的代理，默认端口是 `1086` 。
  - 你可以根据自己需要换成自己的代理配置。

Windows

浏览器

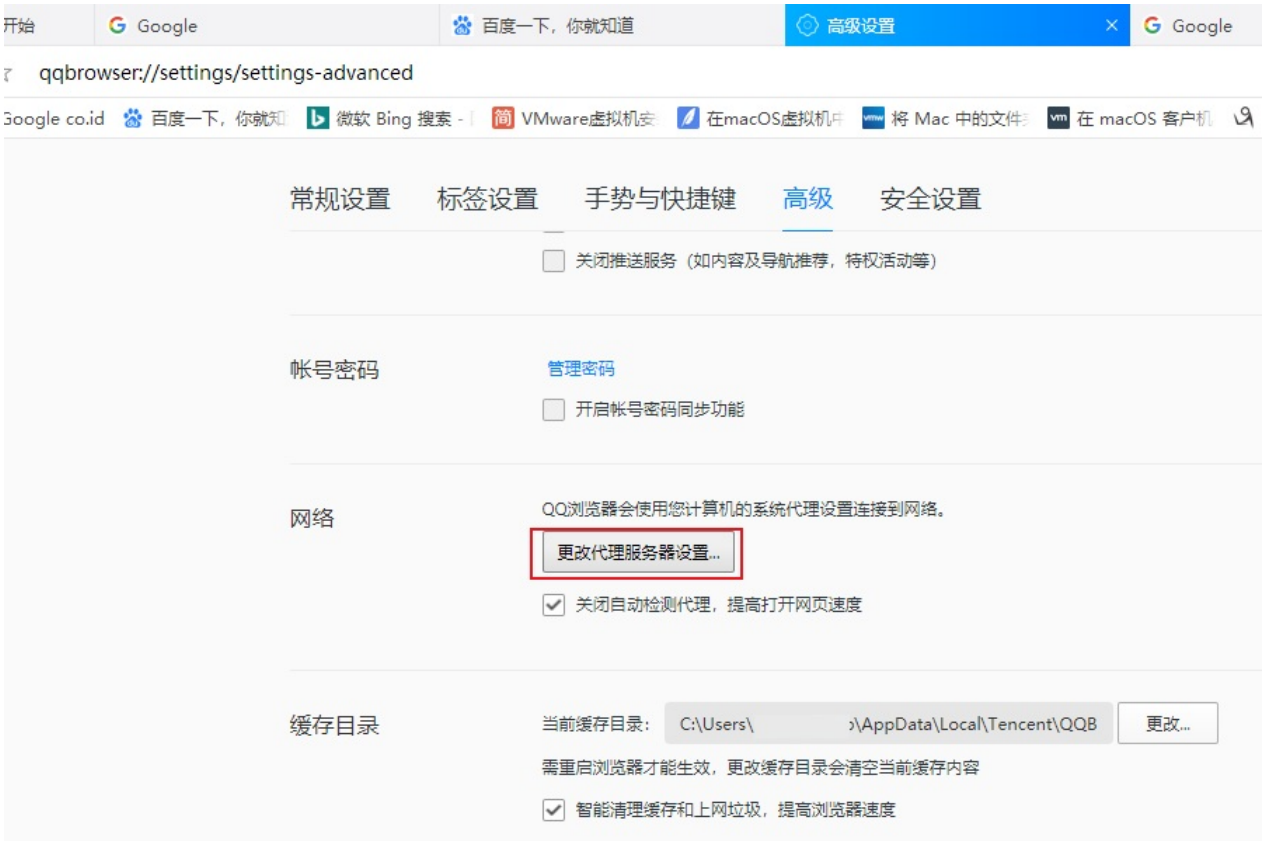
QQ浏览器

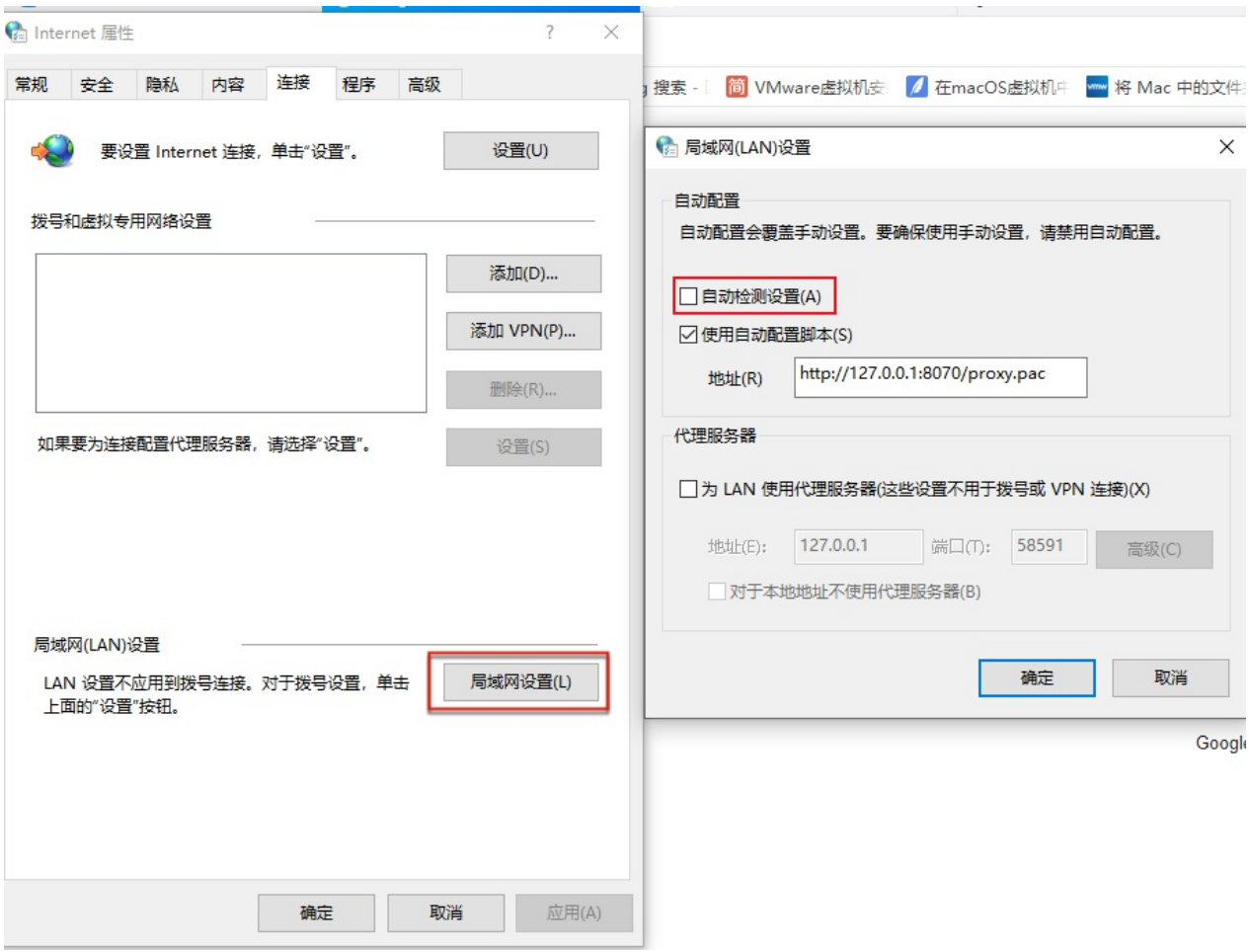
win10中给QQ浏览器加上自动代理

代理设置

qqbrowser://settings/settings-advanced

-》网络 更改代理服务器配置





相关内容：WiFi网络已连接，但是QQ浏览器无法上网

- **原因**：代理配置的问题
- **解决办法**：去掉代理，即可上网
  - 具体步骤：QQ浏览器-》设置-》高级-》网络-》更改代理服务器设置-》（系统的）Internet属性-》连接-》局域网LAN设置-》局域网设置-》代理服务器-》取消勾选：为LAN使用代理服务器-》确定
  - 注：
    - 之前自己的本地代理的软件Trojan，不论是否启动，是否开启或关闭，都不会影响QQ浏览器上网的。
    - 此处不知道，是什么程序，软件，何时何地，去改了此处的系统的代理，导致无法上网
    - 所以根本原因：未知

附录

下面列出相关参考资料。

参考资料

- 最流行的版本管理系统：Git
- 【已解决】Mac中给pip3添加代理以提升下载python包的速度 – 在路上
- 【已解决】用Python代码测试多贝云代理IP是否生效
-
- 【已解决】小米9安卓手机中安装Charles证书：没有可用连接，因为代理服务器故障或其地址有误
- 【已解决】给Python的requests加上代理访问WordPress的REST的api
- 【已解决】iOS自动化安装app：给当前WiFi去掉代理以及自动安装app后再恢复之前代理
- 【整理】后续可能会用到的其他各家的IP代理
- 【记录】购买阿布云的每次请求IP都不同的动态IP代理
- 【已解决】如何破解大众点评网页爬取时的反扒验证verify.meituan.com
- 【已解决】PySpider中使用多贝云IP代理池实现每次请求IP都不同
- 【已解决】购买多贝云IP代理池
- 【已解决】用Python代码测试多贝云代理IP是否生效
- 【已解决】找个好用的IP代理池实现防止大众点评网站的反扒
- 【已解决】搞懂IP代理池相关概念和逻辑
- 【已解决】mac中给brew设置代理加速下载和更新
- 【已解决】锤子手机M1L设置WiFi网络代理
- 【已解决】给iPhone中设置Charles的Wifi代理
- 【已解决】给自动抓包工具安装Python虚拟环境
- 【未解决】Mac中给pip更换源以加速下载
- 【已解决】Windows中Trojan无法上网：错误代码 ERR_CONNECTION_RESET
- 【已解决】Win中已经连上WiFi网络但是无法上网
- 【已解决】Android Studio中给AS本身设置代理
- 【已解决】Android Studio中给Gradle添加设置代理
- 【已解决】给用sshpass的rsync加代理以加速
- 【已解决】Win10中给Python的pip安装库加代理提高下载速度
- 【部分解决】Mac中给git添加加一次的当前的临时代理
- 【已解决】mac中给git加代理加速git clone下载代码的速度
- 【已解决】mac中git push只对github用代理而对gitee不用代理
- 【已解决】Mac中SecureCRT通过ssh访问Vultr的VPS服务器太慢
-
- 代理技术简介_Thehope way - SegmentFault 思否
- 代理服务器 - 维基百科，自由的百科全书
- 代理IP广泛的技术应用在哪？-万变ip
-