

Proxypool代理池搭建

2023-04-09 21:18:23 本文共 5600 字 阅读完需 22.5 分钟

该项目是基于python爬虫制作的定时获取免费可用代理并入池的代理池项目。本文将来具体实现一下相关操作。

前言

项目地址：https://github.com/jhao104/proxy_pool

这个项目是github上一个大佬基于python爬虫制作的定时获取免费可用代理并入池的代理池项目

我们来具体实现一下。

具体操作

1.安装配置redis

将自动爬取的代理入池需要redis数据库，首先就得安装redis。

redis官方建议我们在linux上安装，安装方式主要有两种，直接包获取或手动安装。

- 指令安装

```
apt-get install redis-server
```

- 手动安装

在[官网](#)下载最新redis安装包，导入Linux。

```
tar -zxvf redis-6.2.6.tar.gz
cd redis-6.2.6/
make
make install
cd /usr/local/bin
mkdir config
cp /opt/redis-6.2.6/redis.conf config          # 默认安装位置
为/opt
```

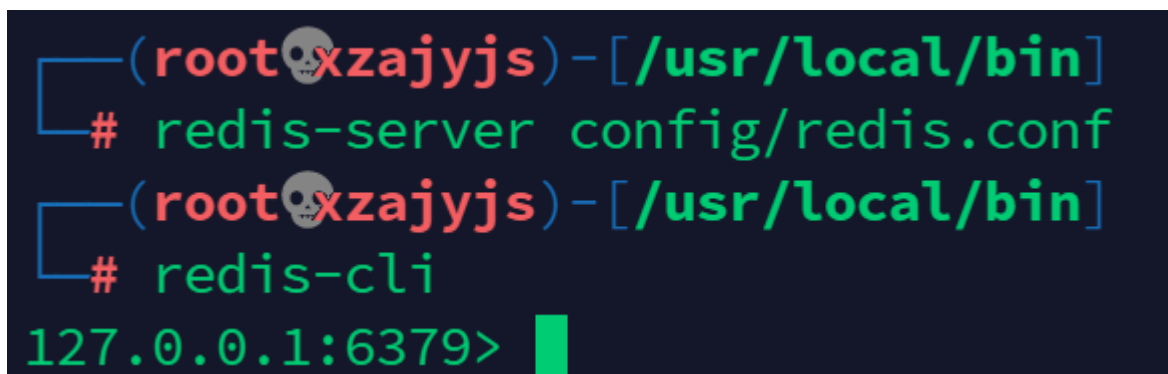
配置文件修改

修改redis配置文件(注意两种安装方式的配置文件位置不同,自动安装在 `/etc/redis/redis.conf` , 手动安装在 `/opt/redis-6.2.6/redis.conf`), 进行如下修改:

```
daemonize yes          # 守护进程开启
protected-mode no      # 关闭保护模式
# bind 127.0.0.1 ::1    # 此条为仅允许本地访问,
                        # 必须注释掉
port 6379               # redis 开放端口(如果是有防火墙的
                        # 服务器需要开启该端口)
```

开启redis

```
redis-server config/redis.conf
redis-cli
```



```
(root@xzajyjs)-[/usr/local/bin]
# redis-server config/redis.conf
(redis@xzajyjs)-[/usr/local/bin]
# redis-cli
127.0.0.1:6379>
```

如需停止:

shutdown

exit

2.拉取并使用脚本

根据项目文档，可以手动配置也可以使用docker部署(推荐)

docker 使用方法见另一篇[博客](#)

```
docker pull jhao104/proxy_pool
```

```
docker run --env DB_CONN=redis://:[password]@[ip]:[port]/[db] -p  
5010:5010 jhao104/proxy_pool:latest
```

password 没有可为空

db 默认0

运行成功应如图:

```
2021-10-19 22:47:15,128 check.py[line:111] INFO UseProxyCheck - thread_01: 112.6.117.135:8085 pass
2021-10-19 22:47:15,131 check.py[line:90] INFO UseProxyCheck - thread_01: complete
2021-10-19 22:47:15,167 check.py[line:111] INFO UseProxyCheck - thread_04: 222.74.202.233:8081 pass
2021-10-19 22:47:15,177 check.py[line:90] INFO UseProxyCheck - thread_04: complete
2021-10-19 22:47:15,261 check.py[line:111] INFO UseProxyCheck - thread_18: 222.74.202.233:80 pass
2021-10-19 22:47:15,267 check.py[line:90] INFO UseProxyCheck - thread_18: complete
2021-10-19 22:47:15,375 check.py[line:111] INFO UseProxyCheck - thread_00: 222.74.202.230:8080 pass
2021-10-19 22:47:15,378 check.py[line:90] INFO UseProxyCheck - thread_00: complete
2021-10-19 22:47:15,414 check.py[line:111] INFO UseProxyCheck - thread_08: 113.96.223.239:4004 pass
2021-10-19 22:47:15,424 check.py[line:90] INFO UseProxyCheck - thread_08: complete
2021-10-19 22:47:15,444 check.py[line:111] INFO UseProxyCheck - thread_11: 125.73.131.137:9091 pass
2021-10-19 22:47:15,450 check.py[line:90] INFO UseProxyCheck - thread_11: complete
2021-10-19 22:47:15,482 check.py[line:111] INFO UseProxyCheck - thread_19: 112.6.117.178:8085 pass
2021-10-19 22:47:15,486 check.py[line:90] INFO UseProxyCheck - thread_19: complete
2021-10-19 22:47:15,634 check.py[line:111] INFO UseProxyCheck - thread_13: 182.92.0.222:8000 pass
2021-10-19 22:47:15,638 check.py[line:90] INFO UseProxyCheck - thread_13: complete
2021-10-19 22:47:15,725 check.py[line:111] INFO UseProxyCheck - thread_10: 113.238.142.208:3128 pass
2021-10-19 22:47:15,727 check.py[line:90] INFO UseProxyCheck - thread_10: complete
2021-10-19 22:47:15,859 check.py[line:111] INFO UseProxyCheck - thread_02: 183.47.237.251:80 pass
2021-10-19 22:47:15,862 check.py[line:90] INFO UseProxyCheck - thread_02: complete
2021-10-19 22:47:15,907 check.py[line:111] INFO UseProxyCheck - thread_12: 111.43.79.98:8085 pass
2021-10-19 22:47:15,912 check.py[line:90] INFO UseProxyCheck - thread_12: complete
2021-10-19 22:47:15,968 check.py[line:111] INFO UseProxyCheck - thread_15: 60.205.158.246:3128 pass
2021-10-19 22:47:15,972 check.py[line:90] INFO UseProxyCheck - thread_15: complete
2021-10-19 22:47:18,411 check.py[line:111] INFO UseProxyCheck - thread_09: 47.100.14.22:9006 pass
2021-10-19 22:47:18,414 check.py[line:90] INFO UseProxyCheck - thread_09: complete
2021-10-19 22:47:21,305 check.py[line:111] INFO UseProxyCheck - thread_17: 47.108.154.194:8080 pass
2021-10-19 22:47:21,308 check.py[line:90] INFO UseProxyCheck - thread_17: complete
2021-10-19 22:47:23,751 check.py[line:117] INFO UseProxyCheck - thread_06: 36.255.211.1:54623 fail, count 1 delete
2021-10-19 22:47:23,755 check.py[line:90] INFO UseProxyCheck - thread_06: complete
2021-10-19 22:47:23,766 check.py[line:117] INFO UseProxyCheck - thread_05: 47.111.18.39:8088 fail, count 1 delete
2021-10-19 22:47:23,768 check.py[line:90] INFO UseProxyCheck - thread_05: complete
2021-10-19 22:47:23,829 check.py[line:117] INFO UseProxyCheck - thread_14: 103.199.84.121:8080 fail, count 1 delete
2021-10-19 22:47:23,831 check.py[line:90] INFO UseProxyCheck - thread_14: complete
2021-10-19 22:47:24,367 check.py[line:111] INFO UseProxyCheck - thread_07: 39.96.25.191:8090 pass
2021-10-19 22:47:24,371 check.py[line:90] INFO UseProxyCheck - thread_07: complete
2021-10-19 22:47:26,337 check.py[line:111] INFO UseProxyCheck - thread_16: 47.242.230.213:12345 pass
2021-10-19 22:47:26,339 check.py[line:90] INFO UseProxyCheck - thread_16: complete
2021-10-19 22:47:33,713 check.py[line:111] INFO UseProxyCheck - thread_03: 117.69.170.55:4216 pass
2021-10-19 22:47:33,715 check.py[line:90] INFO UseProxyCheck - thread_03: complete
```

3.生成配置文件并导入Proxyfier

首先pip安装redis包

```
pip install redis
```

编译以下代码，注意修改第8行的ip和port（redis）

```
# -*- coding:utf8 -*-
import redis
import json
from xml.etree import ElementTree

def RedisProxyGet():
    ConnectString = []
    pool = redis.ConnectionPool(host='[ip]', port=[port], db=0,
decode_responses=True)
    use_proxy = redis.Redis(connection_pool=pool)
    key = use_proxy.hkeys('use_proxy')
    for temp in key:
        try:

ConnectString.append(json.loads(use_proxy.hget('use_proxy',temp)
))

        except json.JSONDecodeError: # JSON解析异常处理
            pass
    return ConnectString

def xmlOutputs(data):
    i = 101
    ProxyIDList = []
    ProxifierProfile = ElementTree.Element("ProxifierProfile")
    ProxifierProfile.set("version", str(i))
    ProxifierProfile.set("platform", "Windows")
    ProxifierProfile.set("product_id", "0")
    ProxifierProfile.set("product_minver", "310")
    Options = ElementTree.SubElement(ProxifierProfile,
"Options")
```

```

Resolve = ElementTree.SubElement(Options, "Resolve")
AutoModeDetection = ElementTree.SubElement(Resolve,
"AutoModeDetection")
AutoModeDetection.set("enabled", "false")
ViaProxy = ElementTree.SubElement(Resolve, "ViaProxy")
ViaProxy.set("enabled", "false")
TryLocalDnsFirst = ElementTree.SubElement(ViaProxy,
"TryLocalDnsFirst")
TryLocalDnsFirst.set("enabled", "false")
ExclusionList = ElementTree.SubElement(Resolve,
"ExclusionList")
ExclusionList.text = "%ComputerName%; localhost; *.local"
Encryption = ElementTree.SubElement(Options, "Encryption")
Encryption.set("mode", 'basic')
Encryption = ElementTree.SubElement(Options,
"HttpProxiesSupport")
Encryption.set("enabled", 'true')
Encryption = ElementTree.SubElement(Options,
"HandleDirectConnections")
Encryption.set("enabled", 'false')
Encryption = ElementTree.SubElement(Options,
"ConnectionLoopDetection")
Encryption.set("enabled", 'true')
Encryption = ElementTree.SubElement(Options,
"ProcessServices")
Encryption.set("enabled", 'false')
Encryption = ElementTree.SubElement(Options,
"ProcessOtherUsers")
Encryption.set("enabled", 'false')
ProxyList = ElementTree.SubElement(ProxifierProfile,
"ProxyList")
for temp in data:
    i += 1 # 从101开始增加
    Proxy = ElementTree.SubElement(ProxyList, "Proxy")
    Proxy.set("id", str(i))
    if not temp['https']:

```

```

        Proxy.set("type", "HTTP")
    else:
        Proxy.set("type", "HTTPS")
        Proxy.text = str(i)
        ProxyIDList.append(i)
    Address = ElementTree.SubElement(Proxy, "Address")
    Address.text = temp['proxy'].split(":", 1)[0]

    Port = ElementTree.SubElement(Proxy, "Port")
    Port.text = temp['proxy'].split(":", 1)[1]

    Options = ElementTree.SubElement(Proxy, "Options")
    Options.text = "48"
    ChainList = ElementTree.SubElement(ProxifierProfile,
"ChainList")

    Chain = ElementTree.SubElement(ChainList, "Chain")
    Chain.set("id", str(i))
    Chain.set("type", "simple")

    Name = ElementTree.SubElement(Chain, "Name")
    Name.text="AgentPool"

    for temp_id in ProxyIDList:
        Proxy = ElementTree.SubElement(Chain, "Proxy")
        Proxy.set("enabled", "true")
        Proxy.text=str(temp_id)
    RuleList = ElementTree.SubElement(ProxifierProfile,
"RuleList")

    Rule = ElementTree.SubElement(RuleList, "Rule")
    Rule.set("enabled", "true")
    Name = ElementTree.SubElement(Rule, "Name")
    Applications = ElementTree.SubElement(Rule, "Applications")
    Action = ElementTree.SubElement(Rule, "Action")

```

```

Name.text="御剑后台扫描工具.exe [auto-created]"
Applications.text="御剑后台扫描工具.exe"
Action.set("type","Direct")

# Rule
Rule = ElementTree.SubElement(RuleList, "Rule")
Rule.set("enabled", "true")
Name = ElementTree.SubElement(Rule,"Name")
Targets = ElementTree.SubElement(Rule,"Targets")
Action = ElementTree.SubElement(Rule,"Action")

Name.text="Localhost"
Targets.text="localhost; 127.0.0.1; %ComputerName%"
Action.set("type", "Direct")

# Rule
Rule = ElementTree.SubElement(RuleList, "Rule")
Rule.set("enabled", "true")
Name = ElementTree.SubElement(Rule, "Name")
Action = ElementTree.SubElement(Rule, "Action")
Name.text = "Default"
Action.text = "102"
Action.set("type", "Proxy")

tree = ElementTree.ElementTree(ProxifierProfile)
tree.write("ProxifierConf.ppx", encoding="UTF-8",
xml_declaration=True)
if __name__ == '__main__':
    proxy_data = RedisProxyGet()
    xmlOutputs(proxy_data)
    print("ProxifierConf.ppx配置文件创建完成....")

```

编译成功生成 `ProxifierConf.ppx` 文件。双击导入proxifier即可

这里proxifier的版本不能太高，否则会报错，建议3.3.1

地址	/	端口	类型
36.134.91.82		8888	HTTP
43.228.180.60		80	HTTP
45.140.90.108		1...	HTTPS
47.115.156.64		3128	HTTPS
59.124.224.180		4378	HTTPS
101.37.22.188		8085	HTTPS
103.199.84.121		8080	HTTP
106.15.180.94		8080	HTTP
111.43.79.98		8085	HTTP
118.6.117.185		8085	HTTP

添加...

编辑...

移除

检查...

从列表中拖放代理服务器，将其添加到链中。

AgentPool		
<input checked="" type="checkbox"/>	117.141.155.242:53281	HTTPS
<input checked="" type="checkbox"/>	202.46.38.11:8080	HTTPS
<input checked="" type="checkbox"/>	117.69.170.55:4216	HTTPS
<input checked="" type="checkbox"/>	221.224.136.211:35101	HTTPS
<input checked="" type="checkbox"/>	203.91.121.212:3128	HTTPS
<input checked="" type="checkbox"/>	101.37.22.188:8085	HTTPS
<input checked="" type="checkbox"/>	117.157.197.18:3128	HTTPS
<input checked="" type="checkbox"/>	59.124.224.180:4378	HTTPS
<input checked="" type="checkbox"/>	124.70.46.14:3128	HTTPS
<input checked="" type="checkbox"/>	47.115.156.64:3128	HTTPS
<input checked="" type="checkbox"/>	112.06.222.240:4004	HTTPS

创建

类型...

移除

拖放链内的代理服务器来更改顺序。
使用复选框来启用/禁用链内的代理。

确定

取消

[官方主页](#) [下载新版](#) [问题反馈](#) [捐赠支持](#) ❤️

浏览器扩展 Circle 阅读助手排版，版权归 www.freebuf.com 所有