3. Python 端口扫描

```
from socket import socket
from threading import Thread
```

需要用到的模块
定义函数分别为TCP扫描和UDP扫描

```
def TCPscan(port,ip):

    try:
        s = socket(AF_INET, SOCK_STREAM)
        s.settimeout(1)
        c = s.connect_ex((ip, port))
        if c == 0:
            print(f"端口 {port} 状态: open")
        else:
            print(f"端口 {port} 状态: closed")
        s.close()
    except:
        pass
```

尝试连接，设置超时时长为1秒，如果连接成功则返回端口开放反之返回端口关闭
UDP扫描

```
def UDPscan(port,ip):

    try:
        s = socket(AF_INET, SOCK_DGRAM)
        s.settimeout(1)
        s.sendto(b'hello', (ip, port))
        try:
            data, addr = s.recvfrom(1024)
            if data is not None:
                print(f"端口 {port} 状态: open")
        except:
            print(f"端口 {port} 状态: closed or uncharted")
    except:
        pass
```

注意UDP扫描的结果可能并不正确，因为UDP需要服务器进行返回数据包如果服务器没有返回就不能判断端口是否存在，先设置尝试连接，超时为1秒，发送hello，判断如果有数据返回则显示open ，没有则显示closed or uncharted

```
def main():
    parser = optparse.OptionParser("usage%prog -s <欲扫描的网址> -p <端口号>,<端口号2> -t TCP扫描 -u UDP扫描")
    parser.add_option("-s", dest="tgtHost", type="string", help="网址")
    parser.add_option("-p", dest="tgtPort", type="string", help="端口")
    parser.add_option("-t", dest="tgtTCP", action="store_true", help="TCP 扫描")
    parser.add_option("-u", dest="tgtUdp", action="store_true", help="UDP 扫描")
    (options, args) = parser.parse_args()
    tgtHost = options.tgtHost
    tgtPort = options.tgtPort
    tgtTCP = options.tgtTCP
    tgtUdp = options.tgtUdp
    if tgtHost is None or tgtPort is None:
        print(parser.usage) #如果tgtHost或tgtPort为空，则打印帮助信息
        exit(0)
    if tgtTCP is not None:
        for port in tgtPort.split(','):
            t = Thread(target=TCPscan, args=(int(port), tgtHost))
            t.start()
    if tgtUdp is not None:
        for port in tgtPort.split(','):
            t = Thread(target=UDPscan, args=(int(port), tgtHost))
            t.start()
            t.join()
```

设置命令行参数-s加网址 -p加端口(每个端口直接使用英文逗号分隔),-u执行UDP扫描-t执行TCP扫描并且如果端口和网址没填的话显示使用方法，并且为每个扫描创建一个线程线程使用join保持不在他们结束之前结束主线程。

```
if __name__ == '__main__':
    main()
```

运行主程序
笔者在这里判断UDP扫描写的是否成功的方法是，在本地运行一个python文件这个

文件的代码如下

```python
# coding = utf-8
from socket import *

s = socket(AF_INET, SOCK_DGRAM)  # AF_INET:ipv4, SOCK_DGRAM:UDP
s.bind(("127.0.0.1", 80))  # 绑定地址和端口
print('等待接收数据...')  # 接收数据
while True:
    recv_data = s.recvfrom(1024)  # 接收数据，1024表示接收的最大数据量
    recv_content = recv_data[0].decode('gbk')
    print(f"接收到的数据是：{recv_data[0].decode('gbk')},from
{recv_data[1]}")
    s.sendto(recv_content.encode('gbk'), recv_data[1])  # 发送数据
    if recv_content == 'exit':
        print('客户端退出')
        break
s.close()
```

就是监听80端口看是否有数据包发送，如果接收到数据包返回他发送的数据，这样UDP扫描就接收到了返回的数据这样就可以判断了(在实战中比较靠运气)
最终扫描器代码

```python
from socket import *
from threading import Thread
import optparse
#ASCII艺术形式打印Qscan
text ='''
欢迎使用Qscan
Qscan是一个简单的端口扫描器
请误使用Qscan进行非法行为
'''
def TCPscan(port,ip):

    try:
        s = socket(AF_INET, SOCK_STREAM)
        s.settimeout(1)
        c = s.connect_ex((ip, port))
        if c == 0:
            print(f"端口 {port} 状态: open")
        else:
            print(f"端口 {port} 状态: closed")
        s.close()
    except:
        pass

def UDPscan(port,ip):

    try:
        s = socket(AF_INET, SOCK_DGRAM)
        s.settimeout(1)
        s.sendto(b'hello', (ip, port))
        try:
            data, addr = s.recvfrom(1024)
            if data is not None:
                print(f"端口 {port} 状态: open")
        except:
            print(f"端口 {port} 状态: closed")
    except:
        pass

def main():
    parser = optparse.OptionParser("usage%prog -s <欲扫描的网址> -p <
端口号>,<端口号2> -t TCP扫描 -u UDP扫描")
    parser.add_option("-s", dest="tgtHost", type="string", help="网
址")
    parser.add_option("-p", dest="tgtPort", type="string", help="端
口")
    parser.add_option("-t", dest="tgtTCP", action="store_true",
help="TCP 扫描")
    parser.add_option("-u", dest="tgtUdp", action="store_true",
help="UDP 扫描")
    (options, args) = parser.parse_args()
    tgtHost = options.tgtHost
    tgtPort = options.tgtPort
    tgtTCP = options.tgtTCP
    tgtUdp = options.tgtUdp
    if tgtHost is None or tgtPort is None:
        print(parser.usage) #如果tgtHost或tgtPort为空，则打印帮助信息
        exit(0)
    if tgtTCP is not None:
        for port in tgtPort.split(','):
            t = Thread(target=TCPscan, args=(int(port), tgtHost))
            t.start()
    if tgtUdp is not None:
        for port in tgtPort.split(','):
            t = Thread(target=UDPscan, args=(int(port), tgtHost))
```

```python
                t.start()
                t.join()
if __name__ == '__main__':
    main()
    print(text)
```