扫描器之目标端口扫描与系统指纹分析

一、实验介绍

1.1 实验内容

利用python的socket模块连接端口-俗称端口扫描，通过对应的端口返回出对应的端口服务。

1.2 实验知识点

- Socket
- 对应端口对应服务
- 多线程的操作
- 扫描器中的使用

1.3 实验环境

- Python2.7
- Xfce终端
- Sublime

1.4 适合人群

本课程难度为一般，属于初级级别课程，适合具有Python基础的用户，熟悉python基础知识加深巩固。

1.5 代码获取

你可以通过下面命令将代码下载到实验楼环境中，作为参照对比进行学习。

```
$ wget http://labfile.oss.aliyuncs.com/courses/761/shiyanlouscan6.zip
$ unzip shiyanlouscan6.zip
```

二、实验原理

在渗透测试的初步阶段通常我们都需要对攻击目标进行信息搜集，而端口扫描就是信息搜集中至关重要的一个步骤。通过端口扫描我们可以了解到目标主机都开放了哪些服务，甚至能根据服务猜测可能存在某些漏洞。

TCP端口扫描一般分为以下几种类型：

1. TCP connect扫描：也称为全连接扫描，这种方式直接连接到目标端口，完成了TCP三次握手的过程，这种方式扫描结果比较准确，但速度比较慢而且可轻易被目标系统检测到。
2. TCP SYN扫描：也称为半开放扫描，这种方式将发送一个SYN包，启动一个TCP会话，并等待目标响应数据包。如果收到的是一个RST包，则表明端口是关闭的，而如果收到的是一个SYN/ACK包，则表示相应的端口是打开的。
3. Tcp FIN扫描：这种方式发送一个表示拆除一个活动的TCP连接的FIN包，让对方关闭连接。如果收到了一个RST包，则表明相应的端口是关闭的。
4. TCP XMAS扫描：这种方式通过发送PSH、FIN、URG、和TCP标志位被设为1的数据包。如果收到了一个RST包，则表明相应的端口是关闭的。

三、实验步骤

3.1 简单的扫描

引入skcket模块中的connect，可以连接一个指定端口。

```
from socket import *

def portScanner(host,port):
    try:
        s = socket(AF_INET,SOCK_STREAM)
        s.connect((host,port))
        print('[+] %d open' % port)
        s.close()
    except:
        print('[-] %d close' % port)
```

3.2 对应端口服务

一般查找对应指纹的方式是先连接到目标端口，然后发送一个指令，根据返回的数据得到对应的指纹，这个方法比较准确但要制作起来非常麻烦，要一个个测试每个端口对应的服务。
这里我提供一个比较简单但容错率比较低的方法，就是每个端口对应一个服务，如果扫描到这个端口，那么端口对应的服务也是这个。对于一般的网站来说，网站管理员一般也不会管理这些端口，不会特意修改，所以还是比较有用的，这里收集了端口服务指纹如下：

```
PORT = {80:"web",8080:"web",3311:"kangle",3312:"kangle",3389:"mstsc",4440:"rundeck",5672:"rabbitMQ",5900:"vnc",60
82:"varnish",7001:"weblogic",8161:"activeMQ",8649:"ganglia",9000:"fastcgi",9090:"ibm",9200:"elasticsearch",9300:"
asticsearch",9999:"amg",10050:"zabbix",11211:"memcache",27017:"mongodb",28017:"mondodb",3777:"dahua jiankong",5
0000:"sap netweaver",50060:"hadoop",50070:"hadoop",21:"ftp",22:"ssh",23:"telnet",25:"smtp",53:"dns",123:"ntp",16
1:"snmp",8161:"snmp",162:"snmp",389:"ldap",443:"ssl",512:"rlogin",513:"rlogin",873:"rsync",1433:"mssql",1080:"so
cks",1521:"oracle",1900:"bes",2049:"nfs",2601:"zebra",2604:"zebra",2082:"cpanle",2083:"cpanle",3128:"squid",3312:
quid",3306:"mysql",4899:"radmin",8834:'nessus',4848:'glashfish'}
```

3.3 代码编写

在 /lib/core 中 编写 PortScan.py

整个代码如下：

```
#!/usr/bin/env python
# __author__= 'w8ay'

import socket
import threading
import Queue

class PortScan:
    def __init__(self,ip="localhotst",threadNum = 5):
        self.PORT = {80:"web",8080:"web",3311:"kangle",3312:"kangle",3389:"mstsc",4440:"rundeck",5672:"rabbitMQ",5
900:"vnc",6082:"varnish",7001:"weblogic",8161:"activeMQ",8649:"ganglia",9000:"fastcgi",9090:"ibm",9200:"elastic
search",9300:"elasticsearch",9999:"amg",10050:"zabbix",11211:"memcache",27017:"mongodb",28017:"mondodb",3777:"d
ahua jiankong",50000:"sap netweaver",50060:"hadoop",50070:"hadoop",21:"ftp",22:"ssh",23:"telnet",25:"smtp",53:"dn
123:"ntp",161:"snmp",8161:"snmp",162:"snmp",389:"ldap",443:"ssl",512:"rlogin",513:"rlogin",873:"rsync",1433:"mss
ql",1080:"socks",1521:"oracle",1900:"bes",2049:"nfs",2601:"zebra",2604:"zebra",2082:"cpanle",2083:"cpanle",3128:"
uid",3312:"squid",3306:"mysql",4899:"radmin",8834:'nessus',4848:'glashfish'}
        self.threadNum = threadNum
        self.q = Queue.Queue()
        self.ip = ip
        for port in self.PORT.keys():
            self.q.put(port)

    def _th_scan(self):
        while not self.q.empty():
            port = self.q.get()
            s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
            s.settimeout(1)
            try:
                s.connect((self.ip, port))
                print "%s:%s OPEN [%s]"%(self.ip,port,self.PORT[port])
            except:
                print "%s:%s Close"%(self.ip,port)
            finally:
                s.close()

    def work(self):
        threads = []
        for i in range(self.threadNum):
            t = threading.Thread(target=self._th_scan())
            threads.append(t)
            t.start()
        for t in threads:
            t.join()
        print('[*] The scan is complete!')
```

在具体实现过程中，首先用队列压入所有要检测的端口，

```
def __init__(self,ip="localhotst",threadNum = 5):
    self.PORT =
{80:"web",8080:"web",3311:"kangle",3312:"kangle",3389:"mstsc",4440:"rundeck",5672:"rabbitMQ",5900:"vnc",6082:"va
rnish",7001:"weblogic",8161:"activeMQ",8649:"ganglia",9000:"fastcgi",9090:"ibm",9200:"elasticsearch",9300:"elas
ticsearch",9999:"amg",10050:"zabbix",11211:"memcache",27017:"mongodb",28017:"mondodb",3777:"dahua jiankong",500
00:"sap
netweaver",50060:"hadoop",50070:"hadoop",21:"ftp",22:"ssh",23:"telnet",25:"smtp",53:"dns",123:"ntp",161:"snmp",8
161:"snmp",162:"snmp",389:"ldap",443:"ssl",512:"rlogin",513:"rlogin",873:"rsync",1433:"mssql",1080:"socks",1521:"
acle",1900:"bes",2049:"nfs",2601:"zebra",2604:"zebra",2082:"cpanle",2083:"cpanle",3128:"squid",3312:"squid",3306:
ysql",4899:"radmin",8834:'nessus',4848:'glashfish'}
    self.threadNum = threadNum
    self.q = Queue.Queue()
    self.ip = ip
    for port in self.PORT.keys():
        self.q.put(port)
```

创建一个线程函数，每个线程调用这个函数，这个函数的功能就是取出队列的端口，然后扫描。

```
def _th_scan(self):
    while not self.q.empty():
        port = self.q.get()
        s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
        s.settimeout(1)
        try:
            s.connect((self.ip, port))
            print "%s:%s OPEN [%s]"%(self.ip,port,self.PORT[port])
        except:
            print "%s:%s Close"%(self.ip,port)
        finally:
            s.close()
```

创建工作函数来调用线程：

```
def work(self):
    threads = []
    for i in range(self.threadNum):
        t = threading.Thread(target=self._th_scan())
        threads.append(t)
        t.start()
    for t in threads:
        t.join()
    print('[*] The scan is complete!')
```

进行测试：

检测实验楼开放的端口情况：

```
115.29.233.149:80 OPEN [web]
115.29.233.149:873 Close
115.29.233.149:1900 Close
115.29.233.149:28017 Close
115.29.233.149:123 Close
115.29.233.149:9090 Close
115.29.233.149:389 Close
115.29.233.149:27017 Close
115.29.233.149:50060 Close
115.29.233.149:3312 Close
115.29.233.149:8080 Close
115.29.233.149:50070 Close
115.29.233.149:1433 Close
115.29.233.149:161 Close
115.29.233.149:162 Close
115.29.233.149:1521 Close
115.29.233.149:443 OPEN [ssl]
115.29.233.149:4848 Close
115.29.233.149:3777 Close
115.29.233.149:6082 Close
115.29.233.149:8649 Close
115.29.233.149:11211 Close
115.29.233.149:4899 Close
115.29.233.149:8161 Close
115.29.233.149:3306 Close
```

3.4 扫描需要的端口扫描器

上面代码保存到 `lib/core/PortScan.py` 文件中，上面代码只能检测单个IP的端口开放情况，但是在扫描器中我们输出的域名，所以我们还需要写一个函数将域名对应到IP上。

3.5 域名->IP

首先我们需要用一个python内置的urlparse模块来解析url。

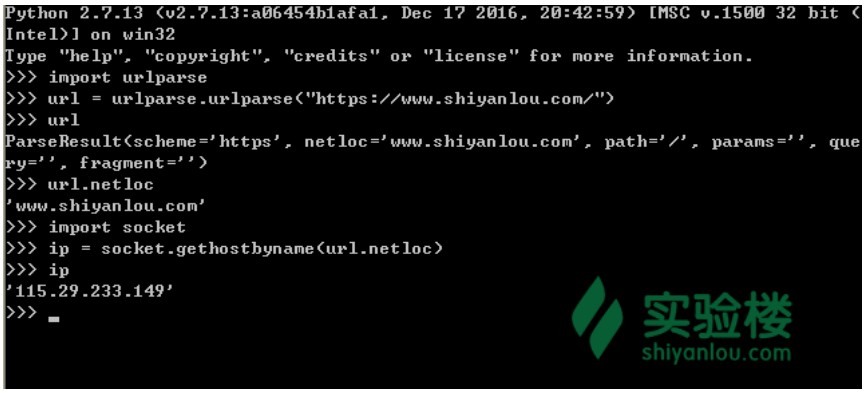
urlparse的简单用法：

```
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:42:59) [MSC v.1500 32 bit (
Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import urlparse
>>> url = urlparse.urlparse("https://www.shiyanlou.com/")
>>> url
ParseResult(scheme='https', netloc='www.shiyanlou.com', path='/', params='', que
ry='', fragment='')
>>>
```

我们需要得到ParseResult中netloc的值即可。

然后用一个 socket.gethostbyname 函数来实现域名到IP的转换。

还是一张图解释用法：



考虑到这个可以写成公共函数，我们就写到 lib/core/common.py 中。

在 common.py 中先导入一些我们需要用的库

```
import urlparse
import socket
```

命名这个函数：

```
def gethostbyname(url):
    domain = urlparse.urlparse(url)
    # domain.netloc
    if domain.netloc is None:
        return None
    ip = socket.gethostbyname(domain.netloc)
    return ip
```

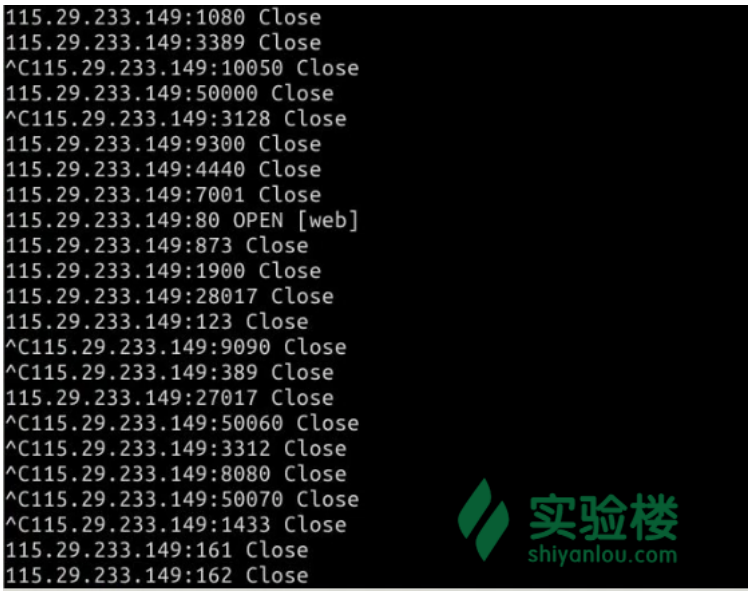
3.6 集成到扫描器中

修改下w8ay.py 这个主入口文件。

```
#!/usr/bin/env python
#-*- coding:utf-8 -*-
'''
Name:w8ayScan
Author:w8ay
Copyright (c) 2017
'''
import sys
from lib.core.Spider import SpiderMain
from lib.core import webcms, common, PortScan


reload(sys)
sys.setdefaultencoding('utf-8')
def main():
    root = "https://shiyanlou.com"
    threadNum = 10
    ip = common.gethostbyname(root)
    print "IP:",ip
    print "Start Port Scan:"
    pp = PortScan.PortScan(ip)
    pp.work()

    #webcms
    ww = webcms.webcms(root,threadNum)
    ww.run()

    #spider
    w8 = SpiderMain(root,threadNum)
    w8.craw()

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

加上我们的端口扫描模块。

最后的结果：



四、实验总结

现在。我们的扫描器运行流程是：

- 域名->转换ip->端口扫描
- CMS识别
- 爬虫信息收集->调用插件

已经有了基本扫描器的雏形了。

上一节 (/courses/761/labs/2654/document)　　下一节 (/courses/761/labs/2671/document)

课程教师

new4
共发布过1门课程

查看老师的所有课程 > (/teacher/102428)