David Parrott

CS 455

Fall 2014

Homework 1 “Simple Attack to a FTP Server”

To run:

./ftpclient\_hw.py

Script should execute properly by calling the filename. If there is a problem locating the correct version of python then remove the first line and call:

python ./ftpclient\_hw.py

The program uses a username and server defined in the header to attempt to login to an FTP server using a password list named ‘rockyou\_light.txt’.

In the first stage, a for loop is used to go through the entire range of possible ports. With each port there is an attempt to connect and a response read from the socket. If the string “FTP” is contained in the response then this is assumed to be the port the server is listening on and the loop is exited.

The contents of rockyou\_light.txt are then read into an array called content. This marks the beginning of stage 2: brute force login. A while loop is used here to check if we’ve gone through the entire list or found a password. Each password is checked using a method tryLogin() which returns a string containing the correct password or None if the test password is not correct. The list is traversed from bottom to top because it was a lot faster.

Once the client is logged in the PASV command is sent to the server and a response is read and used to calculate the port which the server has opened for data. Once in passive mode the client sends the LIST command, beginning stage 3 and displaying the server response message as well as whatever data is sent over the data socket. This response is split and then examined looking for a string that contains both ‘cs455’ and ‘programming’. This is assumed to be the name of the file we are attempting to locate and download, the contents of which are:

“Congratulations! Your mission for programming homework 1 is completed.

Bye!”

PASV is sent a second time in order to open a new data socket. The RETR command is then sent with the filename extracted from the LIST response. The file is written with the same file name as was found on the server. Finally the QUIT command is sent and the socket is closed.

“your opinion on whether FTP should use only one TCP connection for both data and control packets and why.”

Since FTP uses TCP connections it makes good sense to use more than one port for data and control. Not only can you send multiple pieces of data simultaneously you can also send data to multiple clients simultaneously. If FTP did not use multiple connections it would be very complicated to accomplish these things.