CS472 Computer Networks
Fall 2020-2021
Homework Assignment # 4
Due Date: Tuesday, November 17th, 2020 at 6:29pm (CLASS TIME)

NOTE: Assignments must be submitted in electronic format via Drexel Learn (http://learning.drexel.edu). All the work must be original, NO TEAM WORK. Late assignments will not be accepted. Please submit your assignment as your first initial and last name as a zip file with all files needed (e.g. mine would be mkain_hw4.zip).

Objective

Homeworks #2 and #3 asked you to implement a popular network protocol. The goal of this assignment is to consider various security features to help it be more secure.

Part A – Add a server configuration file and restrict PORT/PASV

This part of the assignment is to modify your server to implement reading in a configuration file when you initialize the server. The configuration file should be in a fixed location (hardcoded to a fixed name, for examples /home/mkain/ftpserverd.conf). This configuration file should be of a few ("attribute" = "value" pairs). It should ignore any lines which begin with a pound sign ("#").

You should modify your server to restrict PORT (and EPRT) and PASV (and EPSV) by the configuration file.

```
# port_mode supported (default = no)
port_mode = NO
# pasv_mode supported (default = yes)
pasv_mode = YES
```

It is possible to set both of these attributes to YES. If both are set to NO, then it is a fatal error and the server should print an error message and exit. You should test via a client using both modes and ensure that the correct error messages are returned as the client tries both modes (PORT or EPRT = port mode, PASV or EPSV = passive mode)

Question to be answered:

1. What are the security considerations with port_mode? With pasv_mode? Why would I want one or the other (think about some of the problems that you had with the client and the server – and who calls who)? Think of the conversation between client and server. Think about how NAT changes this – is it a good thing that an application knows about IP addresses?
2. Think about the security implications of a fixed pathname (although probably in a system directory when it would be deployed as a system service) rather than a relative (like out of the current directory). Describe the "it depends" of these two approaches.

Part B - Logging

Question to be answered:

3. Why is logging an important part of security?
4. Do you see any problems with concurrent servers and log files? (dealing with multiple processes or threads writing to the log file at the same time)? How can you solve this problem to keep one logfile for the server even though there are multiple threads/processes trying to write?

Part C – Securing the connection with SSL/TLS

Questions to be answered:

5. What are the different issues with securing the connections with IMPLICIT mode (which is a separate server listening with TLS always at port 990 by default) and EXPLICIT mode (which is the same server waiting on port 21 for special commands to turn TLS on and off)? What are the "it depends" part of the tradeoffs? Think of the data that you're transporting, both on the command channel and data channel.

EXTRA CREDIT (worth up to 20 points) – implement one of the two modes (look at openssl() and other objects for sockets to implement, depending on language).

Part D – Analyzing the conversation

EXTRA CREDIT (worth up to 10 points): Think of the conversation of FTP and compare it to other file transfer protocols

- SFTP – offers the service on port 22 and data and commands share the same channel – better or worse?
- BitTorrent – offers files from a large number of hosts.

What are the good points and bad points of each approach (FTP, SFTP, BitTorrent)?

Part E – Analyzing the operation of the server

Question to be answered:

6. Do you think there are events that you've logged which show that someone is trying to break into your server? Do you have to add other log entries and checking to make sure that attacks aren't happening? Brainstorm some attacks against your FTP server. For at least one of them, implement additional code to check for them and log appropriate entries.

Your submission

Your submission (All client and server code) MUST contain the following:

- Well documented code (VERY WELL documented code) that should compile correctly. Please resubmit ***ALL*** code for the server, not just what was modified.

- A README file detailing instructions to compile your code or the use of your makefile and how to run your code. Please include the name of the compiler and the operating system (and/or system) that you tested it on. Please also include the list of commands that can be used from your UI.
- Server log from a sample run using both your client and the released FTP client.
- The answers to the questions above (and JUSTIFY YOUR ANSWERS!!!!)
- Any other information you deem important (like your name, etc.)

Point Sheet

Item	**Points**
Part A - implementation	30
Part A – question #1	15
Part A – question #2	10
Part B – question #3	10
Part B – question #4	10
Part C – question #5	10
Part E – question #6 and implementation	15
Total	**100**
Part C – extra credit	Up to 20 pts
Part D – extra credit	Up to 10 pts
Grand Total	**130**