Test Script for CNT 4004 Project – Fall 2018

(Christensen – November 21, 2018 – Version 1.02)

The student is expected to bring with them the knock client, knock server, and web server programs in executable and source code form. The student is also expected to bring the completed protocol design document (as described in the project requirements) for submission. It is preferred that student teams bring two laptops with them for their checkout. **Students should contact the instructor or TA if they are unable to bring two laptops for the demo.**

Definitions:

Key terms used in this document are:

- client – PC that will run the knock_client program
- server – PC that will run the knock_server and web_server programs
- knock_client – Program to generate knock
- knock_server – Program to receive and process knock to open web server
- web_server – Web server controlled by knock_server program

System set-up:

One PC connected to the Internet with known IP address to be the "server"

- Have knock_server and web_server installed
- Have three HTML files to be served (see Appendix A)
- Have one large data file to be server (see Appendix A)

One PC connected to the Internet to be the "client"

- Have knock_client installed
- Have a web browser (e.g., Chrome) installed

Test #1 – Basic operation with single knock:

1) Verify that all programs are installed and running
2) Clear browser cache
3) Using browser try to access test1.html on server
   - ~~Should see 404 error displayed on browser~~ Should see that browser cannot connect
4) Execute knock_client
5) Immediately using browser try to access test1.html on server
   - Should see test1.html displayed on browser
6) Wait 10 seconds
7) Using browser try to access test2.html on server
   - ~~Should see 404 error displayed on browser~~ Should see that browser cannot connect
8) Execute knock_client
9) Immediately using browser try to access test3.html on server
   - Should see test3.html displayed on browser

Test #2 – Basic operation with multiple knocks to extend open time:

1) Verify that all programs are installed and running
2) Clear browser cache
3) Using browser try to access test1.html on server
   - ~~Should see 404 error displayed on browser~~ Should see that browser cannot connect
4) Execute knock_client
5) Immediately using browser try to access test1.html on server
   - Should see test1.html displayed on browser
6) Wait 5 seconds
7) Execute knock_client
8) Wait 8 seconds
9) Using browser try to access test2.html on server
   - Should see test2.html displayed on browser
10) Wait 5 seconds
11) Using browser try to access test3.html on server
    - ~~Should see 404 error displayed on browser~~ Should see that browser cannot connect

Test #3 – Basic operation with longer than 10 second file download:

1) Verify that all programs are installed and running
2) Clear browser cache
3) Using browser try to access test1.html on server
   - ~~Should see 404 error displayed on browser~~ Should see that browser cannot connect
4) Execute knock_client
5) Immediately using browser ~~try to~~ access testData.dat
   - ~~Should see test1.html displayed on browser~~
6) Wait 10 seconds
   - Should see incomplete file transfer (verify with file size)

Test #4 – Resistance to DoS attack:

- Using a third PC (TA PC), send UDP packets at a high rate of speed to known ports
- Repeat Test #1
  - The test should complete successfully

Test #5 – Verification that packets meet protocol design:

- By inspection of output of all packet payloads, verify that packet encoding meets that described in protocol design document
- See appendix A for description of "output of all packet payloads"

Test #6 – Verification of support for multiple clients:

- By inspection of source code, verify that multiple clients, each with a different shared secret, can be supported

Test #7 – Verification of unaltered web server (extra credit):

- By inspection verify that knock_server can execute an unaltered web server

---

Document history:

Version 1.01 – Removed two words and one line in test #3, removed text is cross-out. Clearly, there should be no display of an HTML file if a data file (binary) is accessed.

Version 1.02 – Clarified that when the port 80 (or 8080) is not open (e.g., web server process is not running) then browsing the web server should result in connection failed, not 404 error. A 404 error is file coming from a running web server.

Appendix A – File specs and packet output code

test1.html:

```
<html><body>
<h1>This is Test ONE</h1>
</body></html>
```

test2.html:

```
<html><body>
<h1>This is Test TWO</h1>
</body></html>
```

test3.html:

```
<html><body>
<h1>This is Test THREE</h1>
</body></html>
```

testData.dat:

Binary file of sufficient size that download will take about 20 seconds on the testbed. Browser should not recognize the file type and will force it to download. See the below link for code to create a binary file or user specified size.

http://www.csee.usf.edu/~kchriste/tools/genfile.c

Output of packet payloads:

All programs must have a mode (e.g., a verbose mode) where the packet payloads (i.e., the out_buf contents for a `sendto()`) are output to the console in hex for all packets sent. Code similar to the below can do this. For the below, `out_buf[]` is an array of `unsigned char` and `out_buf_len` is the integer-valued length of `out_buf[]`.

```
for (i=0; i<out_buf_len; i++)
  printf("%02X", out_buf[i]);
printf("\n");
```