

8505 Asn 1 - Testing and Usage

How does one dance the Futterwacken?

Isaac Morneau; A00958405

8505 Asn 1 - Testing and Usage	1
README	3
Build and Run	4
Steps	4
Expected Result	4
Run	5
Expected Result	5
Test Cases	6
Unit Tests	6
Manual Testing	8

README

Replicated below as shown on <https://github.com/isaacmorneau/8505-asn1>

README.md

8505-asn1

Building

to keep the root folder orderly it is recommended to make from within bin as follows

```
git clone https://github.com/isaacmorneau/8505-asn1.git
mkdir 8505-asn1/bin
cd bin
cmake ../
make
```

Running

Client usage info:

```
$ ./8505-asn1-client
-[-v]ersion - current version
-[-t]est    - run the test suite if compiled in debug mode
-[-f]ile    - the path to the file to send
-[-p]ort    - the server port to connect to
-[-h]ost    - the host to connect to
-[-d]elay   - seconds between messages
--help      - this message
```

Server usage info:

```
$ ./8505-asn1-server
-[-v]ersion - current version
-[-t]est    - run the test suite if compiled in debug mode
-[-f]ile    - the path to the file to recv
-[-p]ort    - the server port to connect to
--help      - this message
```

Build and Run

Steps

1. Run `cmake ../` inside of ./bin
2. Run `make`

Expected Result

```
18:46:42(master)isaac@HMS-Brixford:bin$ cmake ../
-- The C compiler identification is GNU 8.2.1
-- The CXX compiler identification is GNU 8.2.1
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Looking for pthread.h
-- Looking for pthread.h - found
-- Looking for pthread_create
-- Looking for pthread_create - not found
-- Check if compiler accepts -pthread
-- Check if compiler accepts -pthread - yes
-- Found Threads: TRUE
-- Configuring done
-- Generating done
-- Build files have been written to: /home/isaac/code/8505-asn1/bin
18:46:46(master)isaac@HMS-Brixford:bin$ make
Scanning dependencies of target 8505-asn1-server
[ 8%] Building C object CMakeFiles/8505-asn1-server.dir/src/server.c.o
[16%] Building C object CMakeFiles/8505-asn1-server.dir/src/crc32.c.o
[25%] Building C object CMakeFiles/8505-asn1-server.dir/src/encoder.c.o
[33%] Building C object CMakeFiles/8505-asn1-server.dir/src/network.c.o
[41%] Building C object CMakeFiles/8505-asn1-server.dir/src/test.c.o
[50%] Linking C executable 8505-asn1-server
[50%] Built target 8505-asn1-server
Scanning dependencies of target 8505-asn1-client
[ 58%] Building C object CMakeFiles/8505-asn1-client.dir/src/client.c.o
[ 66%] Building C object CMakeFiles/8505-asn1-client.dir/src/crc32.c.o
[ 75%] Building C object CMakeFiles/8505-asn1-client.dir/src/encoder.c.o
[ 83%] Building C object CMakeFiles/8505-asn1-client.dir/src/network.c.o
[ 91%] Building C object CMakeFiles/8505-asn1-client.dir/src/test.c.o
[100%] Linking C executable 8505-asn1-client
[100%] Built target 8505-asn1-client
18:46:48(master)isaac@HMS-Brixford:bin$ |
```

Run

1. Run `./8505-asn1-client`
2. Run `./8505-asn1-server`

Expected Result

```
18:46:48(master)isaac@HMS-Brixford:bin$ ./8505-asn1-client
-[-v]ersion - current version
-[-t]est    - run the test suite if compiled in debug mode
-[-f]ile    - the path to the file to send
-[-p]ort    - the server port to connect to
-[-h]ost    - the host to connect to
-[-d]elay   - seconds between messages
--help     - this message

18:48:16(master)isaac@HMS-Brixford:bin$ ./8505-asn1-server
-[-v]ersion - current version
-[-t]est    - run the test suite if compiled in debug mode
-[-f]ile    - the path to the file to recv
-[-p]ort    - the server port to connect to
--help     - this message

18:48:59(master)isaac@HMS-Brixford:bin$ |
```

Test Cases

Unit Tests

This project, when compiled in debug mode, includes unit tests for all encoding and transcoding functionality. Each set of tests is overviewed below.

Most basic case of testing the interface

```
18:52:51(master)isaac@HMS-Brixford:bin$ ./8505-asn1-server -t
basic test

==sending>
[message data]
size: 22
data: testing messag
crc32: 0xCE1D0518
[internal data]
index: 22
slice: 2
len: 15
[raw hex]
15 FF F0 74 65 73 74 69 6E 67 20 6D 65 73 73 61 67 65 18 05 1D CE
==recieving>
[message data]
size: 21
data: testing message
crc32: 0x512C1B65
[internal data]
index: 21
slice: 2
len: 15
[raw hex]
15 00 74 65 73 74 69 6E 67 20 6D 65 73 73 61 67 65 18 05 1D CE
[PASSED] crc32 check
```

The encoder is actually designed to be able to handle any size of field as I was uncertain how much space would be available depending on what fields I was planning on using for exfiltration. As such it is able to handle multiple sizes of fields.

```
different field sizes

testing slice of 1 bytes
[PASSED] crc32 check
testing slice of 2 bytes
[PASSED] crc32 check
testing slice of 3 bytes
[PASSED] crc32 check
testing slice of 4 bytes
[PASSED] crc32 check
testing slice of 5 bytes
[PASSED] crc32 check
testing slice of 6 bytes
[PASSED] crc32 check
testing slice of 7 bytes
[PASSED] crc32 check
testing slice of 8 bytes
[PASSED] crc32 check
testing slice of 9 bytes
[PASSED] crc32 check
testing slice of 10 bytes
[PASSED] crc32 check
```

As the frames are designed to be sent over a network one of the tests involves sending it without a preknown size to ensure anything can be sent at run time.

```
testing unknown inbound sizing

[PASSED] crc32 check
[PASSED] outbound finished
[PASSED] inbound finished
```

Finally as the transcoding itself had a need to avoid zeros the functionality is ensured manually just to be certain all transcoding maintains the data and the sizing.

```
tr encoding test
raw data
58 69 00 21 FF FF
[PASSED] tr encoding sizing correct
encoded data
58 69 FF F0 21 FF FF FF FF
decoded data
58 69 00 21 FF FF
[PASSED] tr maintained 0, 0xFF, and regular data
```

Manual Testing

Test	Steps	Result
Send a File	<p>Server Machine</p> <ol style="list-style-type: none"> 1. Run <code>./8505-asn1-server -f test`</code> 2. Wait for frame to print 3. Run <code>`cat test`</code> <p>Client Machine</p> <ol style="list-style-type: none"> 1. Run <code>`echo "hello world" > test`</code> 2. Run <code>./8505-asn-client -f test -h <server address>`</code> 3. Wait for file to send 	<p>Client sends till the file is transfered</p> <p>Server reads the file</p> <p>The file contains "hello world"</p>
Test Daemon Mode	<p>Server Machine</p> <ol style="list-style-type: none"> 1. Run <code>./8505-asn1-server -f test`</code> 2. Wait for frame to print 3. Run <code>`cat test`</code> <p>Client Machine</p> <ol style="list-style-type: none"> 1. Run <code>`echo "hello world" > test`</code> 2. Run <code>./8505-asn-client -f test -h <server address> -b`</code> 3. Client Exits immediately 	<p>Client exits immediately</p> <p>Server reads the file</p> <p>The file contains "hello world"</p>