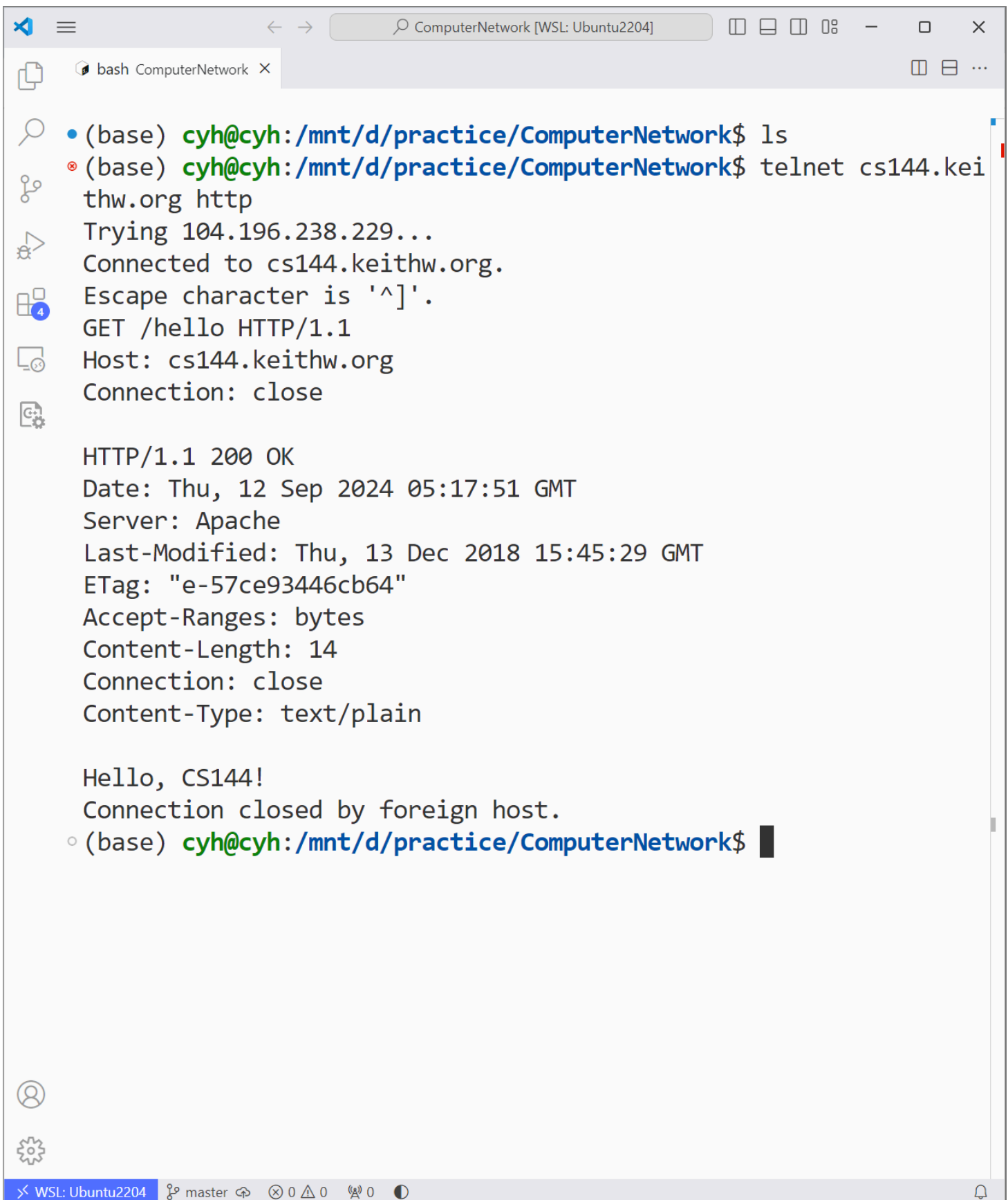


实验0报告

telnet使用

这部分按照实验手册做就完事了，唯一的难点就是为了申请smtp服务，qq邮箱的一堆验证，以及qq邮箱的验证虽然是一堆乱码，但居然不是base64编码的，需要我再编码一遍



The image shows a terminal window titled "ComputerNetwork [WSL: Ubuntu2204]". The terminal output is as follows:

```
(base) cyh@cyh:/mnt/d/practice/ComputerNetwork$ ls
(base) cyh@cyh:/mnt/d/practice/ComputerNetwork$ telnet cs144.keithw.org http
Trying 104.196.238.229...
Connected to cs144.keithw.org.
Escape character is '^]'.
GET /hello HTTP/1.1
Host: cs144.keithw.org
Connection: close

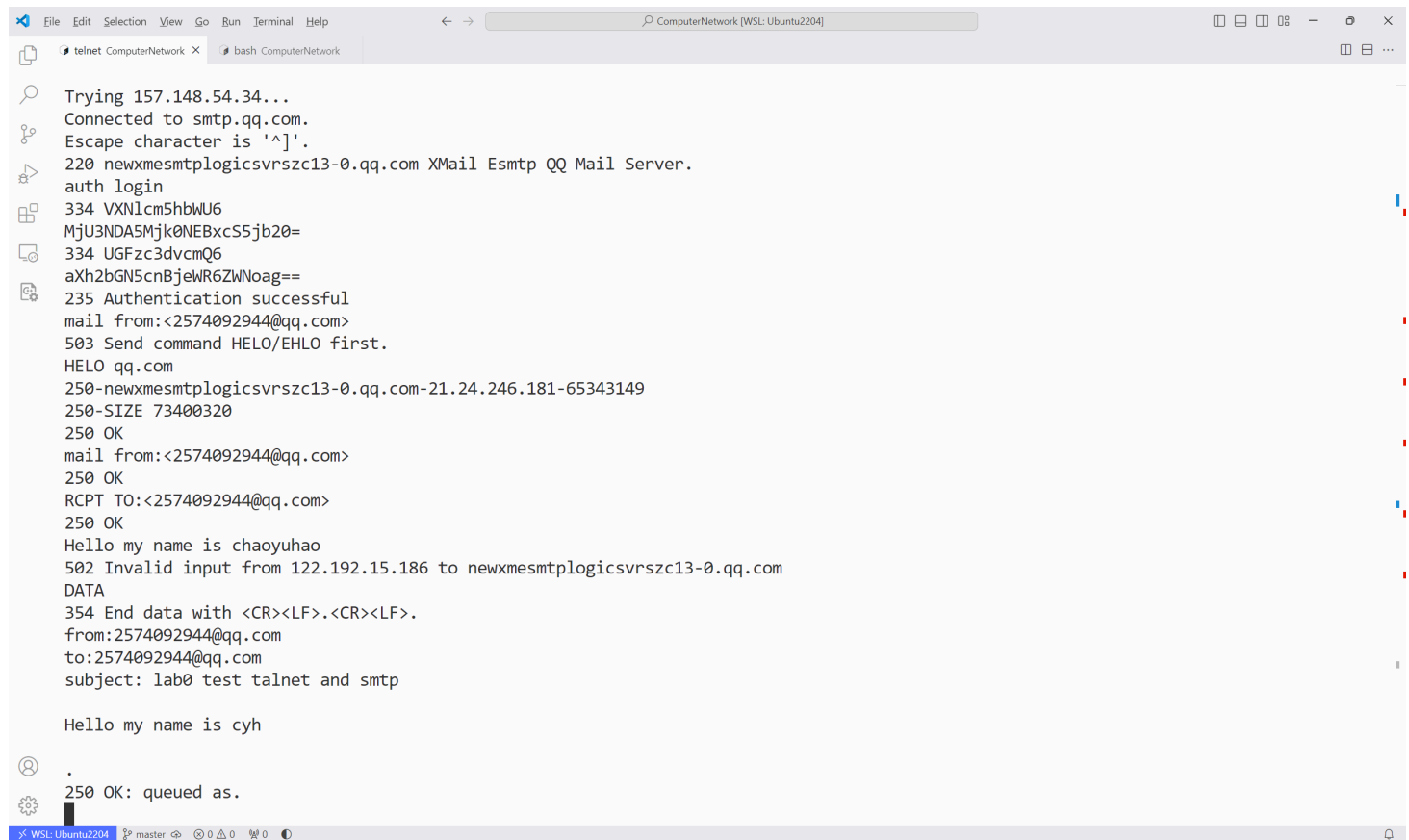
HTTP/1.1 200 OK
Date: Thu, 12 Sep 2024 05:17:51 GMT
Server: Apache
Last-Modified: Thu, 13 Dec 2018 15:45:29 GMT
ETag: "e-57ce93446cb64"
Accept-Ranges: bytes
Content-Length: 14
Connection: close
Content-Type: text/plain

Hello, CS144!
Connection closed by foreign host.
(base) cyh@cyh:/mnt/d/practice/ComputerNetwork$
```

The terminal window includes standard Ubuntu icons on the left and a status bar at the bottom showing "WSL: Ubuntu2204" and "master" branch information.

smtp 发邮件

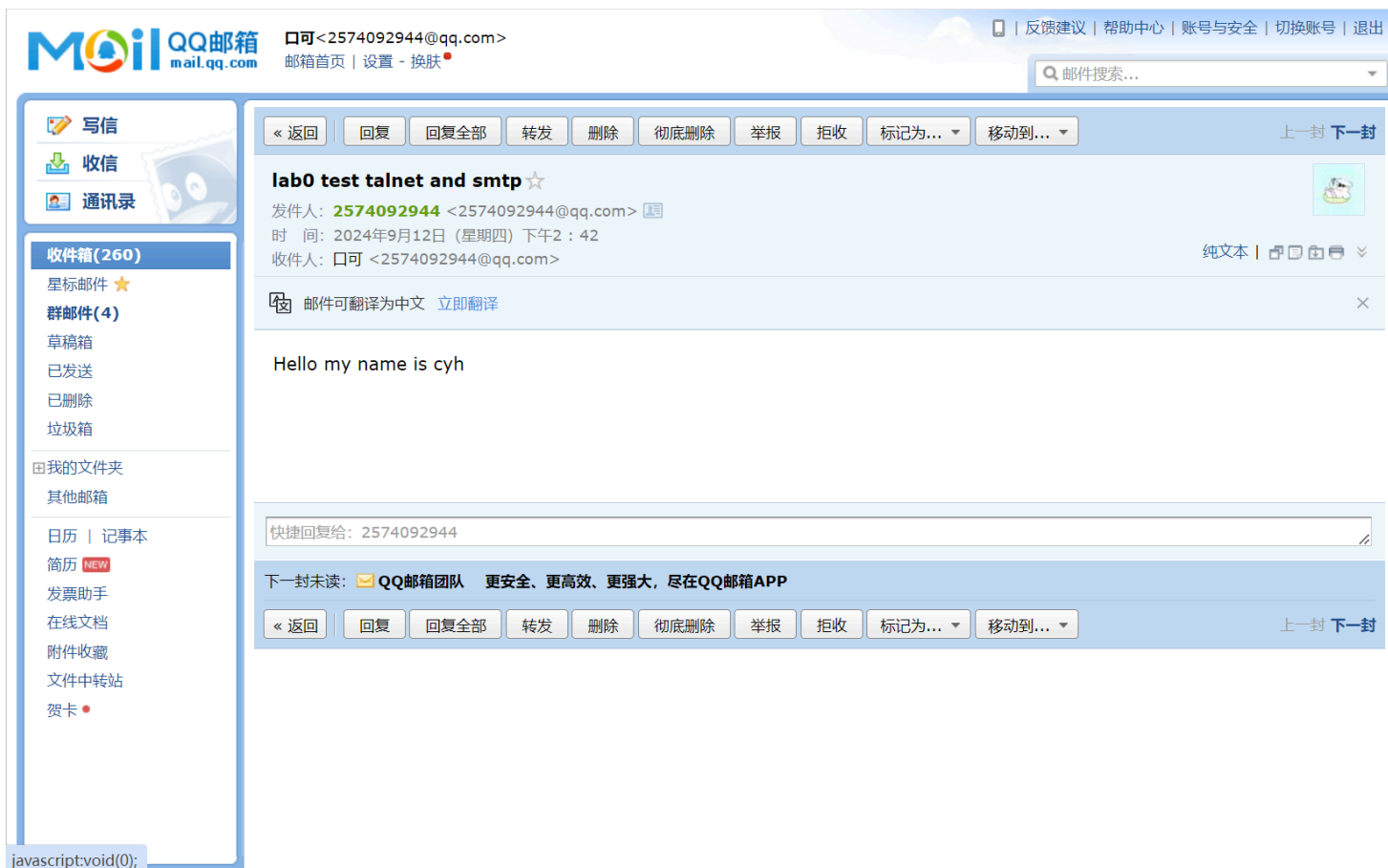
这部分按照实验手册做就完事了，唯二的难点就是为了申请smtp服务，qq邮箱的一堆验证，以及qq邮箱的验证虽然是一堆乱码，但居然不是base64编码的，需要我再编码一遍



```
File Edit Selection View Go Run Terminal Help
ComputerNetwork [WSL: Ubuntu2204]
telnet ComputerNetwork x bash ComputerNetwork

Trying 157.148.54.34...
Connected to smtp.qq.com.
Escape character is '^]'.
220 newxmesmtplogicsvrszc13-0.qq.com XMail Esmtp QQ Mail Server.
auth login
334 VXN1cm5hbWU6
MjU3NDASMjk0NEBxcS5jb20=
334 UGFzc3dvcmQ6
aXh2bGN5c2BjewR6ZWNoag==
235 Authentication successful
mail from:<2574092944@qq.com>
503 Send command HELO/EHLO first.
HELO qq.com
250-newxmesmtplogicsvrszc13-0.qq.com-21.24.246.181-65343149
250-SIZE 73400320
250 OK
mail from:<2574092944@qq.com>
250 OK
RCPT TO:<2574092944@qq.com>
250 OK
Hello my name is chaoyuhao
502 Invalid input from 122.192.15.186 to newxmesmtplogicsvrszc13-0.qq.com
DATA
354 End data with <CR><LF>.<CR><LF>.
from:2574092944@qq.com
to:2574092944@qq.com
subject: lab0 test telnet and smtp

Hello my name is cyh
.
250 OK: queued as.
```



网络通信

```
◦ (base) cyh@cyh:/mnt/d/practice/ComputerNetwork$ netcat -v -l -p 9090
Listening on 0.0.0.0 9090
Connection received on localhost 57552
```

hello

I am netcat

I am telnet



实现t_webget

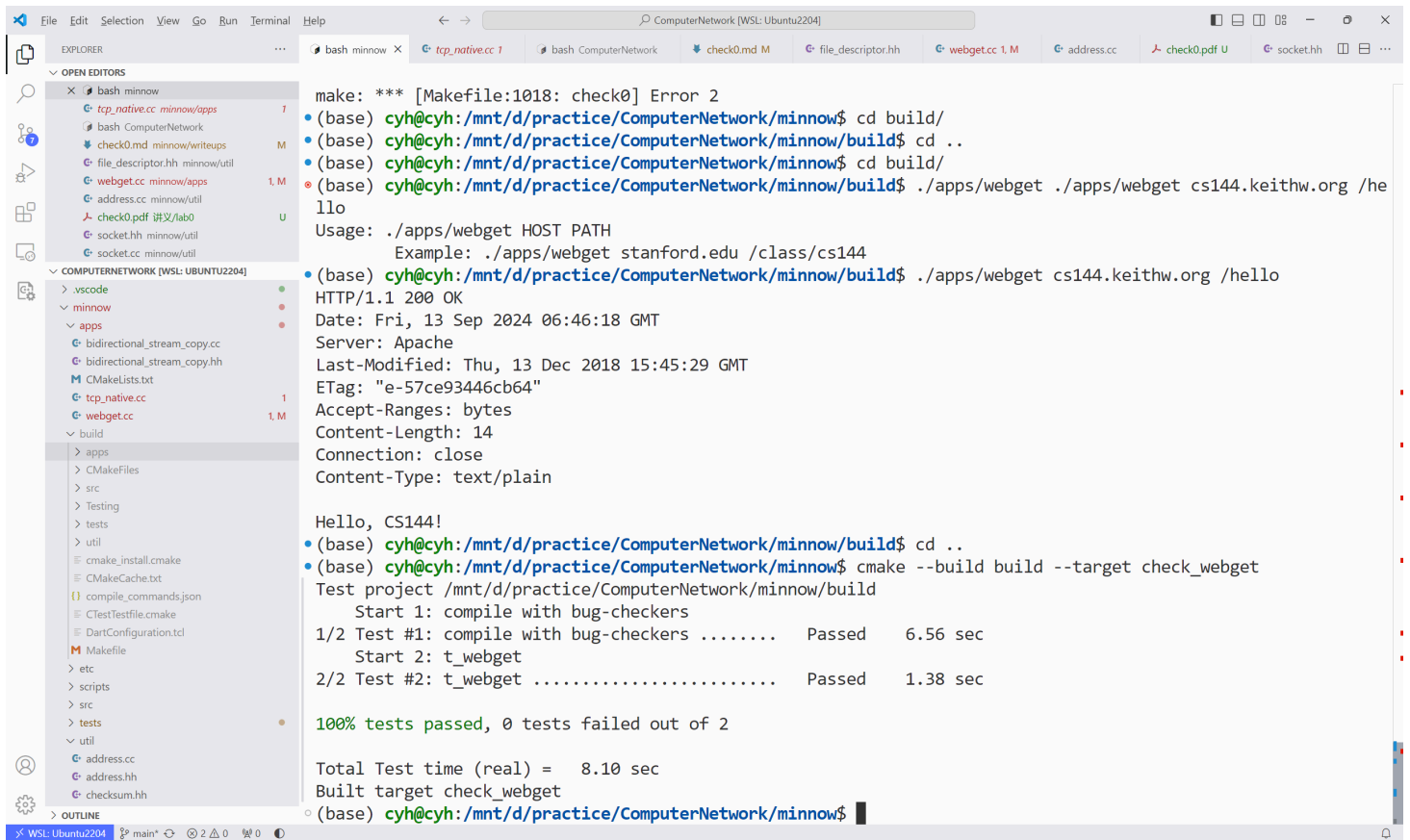
首先这个C++20实在是太先进，我必须先给我的wsl2手动更新cmake，然后又不得不手动更新g++，g++11居然会报错。

这一部分就是复制之前对telnet的使用，需要注意的是调试之前一定要先用telnet试试是不是自己的网真有问题。。。

还有一个重点就是接受字符串，它可能不会一次性发全，需要我們不断地去读，直到EOF。

代码很短，直接贴上来了。

```
void get_URL( const string& host, const string& path )
{
    TCPSocket t_socket;
    Address addr(host, "http");
    t_socket.connect(addr);
    t_socket.write("GET " + path + " HTTP/1.1\r\n");
    t_socket.write("Host: " + host + "\r\n");
    t_socket.write("Connection: close\r\n");
    t_socket.write("\r\n");
    std::string buffer, recive;
    while(buffer.empty()) {
        t_socket.read(buffer);
    }
    while(!buffer.empty()) {
        recive = recive + buffer;
        t_socket.read(buffer);
    }
    cout << recive;
}
```



```
make: *** [Makefile:1018: check0] Error 2
• (base) cyh@cyh:/mnt/d/practice/ComputerNetwork/minnow$ cd build/
• (base) cyh@cyh:/mnt/d/practice/ComputerNetwork/minnow/build$ cd ..
• (base) cyh@cyh:/mnt/d/practice/ComputerNetwork/minnow$ cd build/
• (base) cyh@cyh:/mnt/d/practice/ComputerNetwork/minnow/build$ ./apps/webget ./apps/webget cs144.keithw.org /hello
Usage: ./apps/webget HOST PATH
Example: ./apps/webget stanford.edu /class/cs144
• (base) cyh@cyh:/mnt/d/practice/ComputerNetwork/minnow/build$ ./apps/webget cs144.keithw.org /hello
HTTP/1.1 200 OK
Date: Fri, 13 Sep 2024 06:46:18 GMT
Server: Apache
Last-Modified: Thu, 13 Dec 2018 15:45:29 GMT
ETag: "e-57ce93446cb64"
Accept-Ranges: bytes
Content-Length: 14
Connection: close
Content-Type: text/plain

Hello, CS144!
• (base) cyh@cyh:/mnt/d/practice/ComputerNetwork/minnow/build$ cd ..
• (base) cyh@cyh:/mnt/d/practice/ComputerNetwork/minnow$ cmake --build build --target check_webget
Test project /mnt/d/practice/ComputerNetwork/minnow/build
  Start 1: compile with bug-checkers
1/2 Test #1: compile with bug-checkers ..... Passed    6.56 sec
  Start 2: t_webget
2/2 Test #2: t_webget ..... Passed    1.38 sec

100% tests passed, 0 tests failed out of 2

Total Test time (real) = 8.10 sec
Built target check_webget
• (base) cyh@cyh:/mnt/d/practice/ComputerNetwork/minnow$
```

实现ByteStream

我使用了一个定长的循环数组，数组使用 `std::vector` 在构造函数里确定长度，实现了这个 `ByteStream`，受限于真实的使用场景，我们可以用两个 `uint64_t` 去记录写入读出数据的量，然后关于数组是否为空以及其他相关 `const` 函数的实现都可以非常简单的通过这两数的四则运算得出。传输速度 0.55Gbit/s。

写得不够“现代”，但是好像比用了 `std::deque` 的兄弟们快了不少，lol

```

• (base) cyh@cyh:/mnt/d/practice/ComputerNetwork/minnow$ cmake --build build --target check0
Test project /mnt/d/practice/ComputerNetwork/minnow/build
  Start 1: compile with bug-checkers
1/10 Test #1: compile with bug-checkers ..... Passed    9.45 sec
  Start 2: t_webget
2/10 Test #2: t_webget ..... Passed    1.31 sec
  Start 3: byte_stream_basics
3/10 Test #3: byte_stream_basics ..... Passed    0.09 sec
  Start 4: byte_stream_capacity
4/10 Test #4: byte_stream_capacity ..... Passed    0.10 sec
  Start 5: byte_stream_one_write
5/10 Test #5: byte_stream_one_write ..... Passed    0.10 sec
  Start 6: byte_stream_two_writes
6/10 Test #6: byte_stream_two_writes ..... Passed    0.09 sec
  Start 7: byte_stream_many_writes
7/10 Test #7: byte_stream_many_writes ..... Passed    0.12 sec
  Start 8: byte_stream_stress_test
8/10 Test #8: byte_stream_stress_test ..... Passed    0.34 sec
  Start 37: compile with optimization
9/10 Test #37: compile with optimization ..... Passed    8.62 sec
  Start 38: byte_stream_speed_test
        ByteStream throughput: 0.55 Gbit/s
10/10 Test #38: byte_stream_speed_test ..... Passed    0.21 sec

100% tests passed, 0 tests failed out of 10

Total Test time (real) = 20.64 sec
Built target check0

```