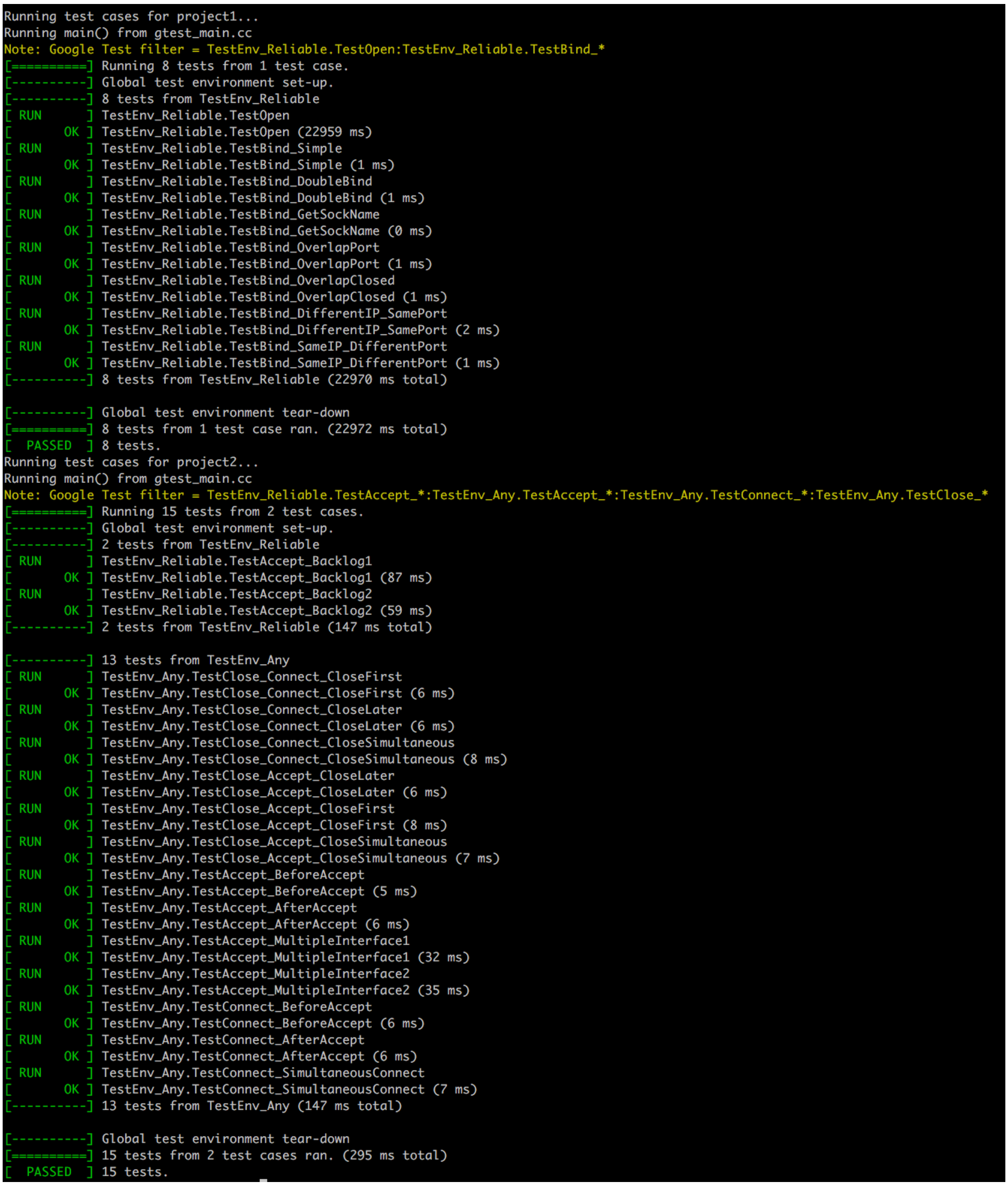
Team30 20140040 Keonil Kim

CS341 Project2-1 Report



```
Running test cases for project1...
Running main() from gtest_main.cc
Note: Google Test filter = TestEnv_Reliable.TestOpen:TestEnv_Reliable.TestBind_*
[==========] Running 8 tests from 1 test case.
[----------] Global test environment set-up.
[----------] 8 tests from TestEnv_Reliable
[ RUN      ] TestEnv_Reliable.TestOpen
[       OK ] TestEnv_Reliable.TestOpen (22959 ms)
[ RUN      ] TestEnv_Reliable.TestBind_Simple
[       OK ] TestEnv_Reliable.TestBind_Simple (1 ms)
[ RUN      ] TestEnv_Reliable.TestBind_DoubleBind
[       OK ] TestEnv_Reliable.TestBind_DoubleBind (1 ms)
[ RUN      ] TestEnv_Reliable.TestBind_GetSockName
[       OK ] TestEnv_Reliable.TestBind_GetSockName (0 ms)
[ RUN      ] TestEnv_Reliable.TestBind_OverlapPort
[       OK ] TestEnv_Reliable.TestBind_OverlapPort (1 ms)
[ RUN      ] TestEnv_Reliable.TestBind_OverlapClosed
[       OK ] TestEnv_Reliable.TestBind_OverlapClosed (1 ms)
[ RUN      ] TestEnv_Reliable.TestBind_DifferentIP_SamePort
[       OK ] TestEnv_Reliable.TestBind_DifferentIP_SamePort (2 ms)
[ RUN      ] TestEnv_Reliable.TestBind_SameIP_DifferentPort
[       OK ] TestEnv_Reliable.TestBind_SameIP_DifferentPort (1 ms)
[----------] 8 tests from TestEnv_Reliable (22970 ms total)

[----------] Global test environment tear-down
[==========] 8 tests from 1 test case ran. (22972 ms total)
[  PASSED  ] 8 tests.
Running test cases for project2...
Running main() from gtest_main.cc
Note: Google Test filter = TestEnv_Reliable.TestAccept_*:TestEnv_Any.TestAccept_*:TestEnv_Any.TestConnect_*:TestEnv_Any.TestClose_*
[==========] Running 15 tests from 2 test cases.
[----------] Global test environment set-up.
[----------] 2 tests from TestEnv_Reliable
[ RUN      ] TestEnv_Reliable.TestAccept_Backlog1
[       OK ] TestEnv_Reliable.TestAccept_Backlog1 (87 ms)
[ RUN      ] TestEnv_Reliable.TestAccept_Backlog2
[       OK ] TestEnv_Reliable.TestAccept_Backlog2 (59 ms)
[----------] 2 tests from TestEnv_Reliable (147 ms total)

[----------] 13 tests from TestEnv_Any
[ RUN      ] TestEnv_Any.TestClose_Connect_CloseFirst
[       OK ] TestEnv_Any.TestClose_Connect_CloseFirst (6 ms)
[ RUN      ] TestEnv_Any.TestClose_Connect_CloseLater
[       OK ] TestEnv_Any.TestClose_Connect_CloseLater (6 ms)
[ RUN      ] TestEnv_Any.TestClose_Connect_CloseSimultaneous
[       OK ] TestEnv_Any.TestClose_Connect_CloseSimultaneous (8 ms)
[ RUN      ] TestEnv_Any.TestClose_Accept_CloseLater
[       OK ] TestEnv_Any.TestClose_Accept_CloseLater (6 ms)
[ RUN      ] TestEnv_Any.TestClose_Accept_CloseFirst
[       OK ] TestEnv_Any.TestClose_Accept_CloseFirst (8 ms)
[ RUN      ] TestEnv_Any.TestClose_Accept_CloseSimultaneous
[       OK ] TestEnv_Any.TestClose_Accept_CloseSimultaneous (7 ms)
[ RUN      ] TestEnv_Any.TestAccept_BeforeAccept
[       OK ] TestEnv_Any.TestAccept_BeforeAccept (5 ms)
[ RUN      ] TestEnv_Any.TestAccept_AfterAccept
[       OK ] TestEnv_Any.TestAccept_AfterAccept (6 ms)
[ RUN      ] TestEnv_Any.TestAccept_MultipleInterface1
[       OK ] TestEnv_Any.TestAccept_MultipleInterface1 (32 ms)
[ RUN      ] TestEnv_Any.TestAccept_MultipleInterface2
[       OK ] TestEnv_Any.TestAccept_MultipleInterface2 (35 ms)
[ RUN      ] TestEnv_Any.TestConnect_BeforeAccept
[       OK ] TestEnv_Any.TestConnect_BeforeAccept (6 ms)
[ RUN      ] TestEnv_Any.TestConnect_AfterAccept
[       OK ] TestEnv_Any.TestConnect_AfterAccept (6 ms)
[ RUN      ] TestEnv_Any.TestConnect_SimultaneousConnect
[       OK ] TestEnv_Any.TestConnect_SimultaneousConnect (7 ms)
[----------] 13 tests from TestEnv_Any (147 ms total)

[----------] Global test environment tear-down
[==========] 15 tests from 2 test cases ran. (295 ms total)
[  PASSED  ] 15 tests.
```

1. Required Functions

accept

(1) Check the validity of input pid, fd.

(2) Check the completeq which stores the established connection waiting for accept to be called.

(1) If empty, add the accept call's information to accept_info_list.

(2) if not, consume one connection and return

connect

(1) Check the validity of input pid, fd

(2) Get ip and port of source(itself) and destination, considering simultaneous open case

(3) construct packet to send and sent it

getpeername

(1) Check the validity of input pid, fd

(2) Get the socket information from estab_list, which stores established connections and return

listen

(1) Check the validity of input pid, fd

(2) Change the corresponding socket's state to 'LISTEN', and initialize the listenq and completeq with empty set and queue for the input pid and fd.

2. Connection setup

I will explain the implementation of packetArrived function.

(1) read the data from arrived packet, swap the source and destination here.

With flag defined in header file, handle each case.

case1(SYNACK)

When SYN and ACK is arrived, it send ACK back.

First, check the validity of connection with the source and destination address information from packet. Also check the validity of ack with seq saved in seq_list. If all happen to be valid, construct the packet with ACK and send it back.

case2(SYN)

When SYN is arrived, it send SYN and ACK back. However, if it happens to be the simultaneous open case, we should send ACK. Otherwise, send back with SYNACK flag, if the arrived packet's data is valid. The validity is checked with the address of source/destination.

case3(ACK)

This case is quite complicated since it should handle backlog and pending connection, and return the blocked accept if required.

As I did in SYN case, the corresponding socket is searched from the bind_list with the address information from the received packet. Of course if invalid, free the packets and return immediately.

When ACK is arrived, we first check whether it is the case of simultaneous open or not. If simultaneous open, we check the cli_list, which stores socket that are not yet established after send connect. It changes the state of the socket to ESTAB and remove it from cli_list and add it to estab_list.

Otherwise, there are two cases.

First, if there is blocked accept, the connection will unblock that accept with the corresponding information from accept_info_list. It stores the UUID of the blocked accept call and two pointer arguments of that call, storing the address of client. I create new fd and create new socket with the address of server and client connected with new fd, and add it to estab_list with state ESTAB. Since the connection is stored in listen queue, remove it from the listen queue to. Then finally return the accept.

Second case, if there is no blocked accept, the connection is established but not waiting for accept to be called and return with new socket. So it is removed from listen queue and moved to complete queue. When accept is called, it checks the complete queue and if not empty, it consumes the pending connection and return with the socket connected to that client.