Lab 2 Writeup

My name: 郑凯琳 TAY KAI LIN

My Student number : 205220025

This lab took me about 10 hours to do. I did attend the lab session.

---

1. Program Structure and Design:

TCP Receiver 会接收到的 3 种报文：

1. SYN 报文

- 包含初始 ISN
- 用于标识字节流的起始位置
- ISN 通常是随机值，防止被攻击

2. FIN 报文

- 表明通信结束

3. 普通的数据报文

- 只需要写入 payload 到 ByteStream 即可

TCP 报文头部的 `seqno` 标识 payload 字节流在完整字节流中的起始位置。 但是 `seqno` 定义为uint32_t类型，最多支持 4gb 字节流，不够。

--> 解决方法：`abs_seqno` (absolute sequence number)，定义为uint64_t类型，支持最多 2^64 - 1 字节流。

3.1 Translating between 64-bit indexes and 32-bit seqnos seqno 和 abs_seqno 转换

思路 & 核心代码：

（1）`abs_seqno` 转换为 `seqno`

- 加法重载
- 对于 `unsigned` 类型，溢出直接归零

Code:

```
WrappingInt32 wrap(uint64_t n, WrappingInt32 isn) {
//DUMMY_CODE(n, isn);
//超出：直接溢出
return WrappingInt32{static_cast<uint32_t>(n)+ isn.raw_value()};
}
```

（2）`seqno` 转换为 `abs_seqno`

- 引入 `checkpoint` ：当前写入的总字节数
- 将 `checkpoint` 从 `abs_seqno` 转换为 `seqno`
  - `wrap` 函数把 `checkpoint` 变为 `WrappingInt32`
- 计算 `n` 和 `checkpoint` 之间的最小步数
  - 到达新报文的步数
  - 减法重载
- 将步数加到 `checkpoint` 上
  - 如果往反方向，结果可能是负数

Code:

```
uint64_t unwrap(WrappingInt32 n, WrappingInt32 isn, uint64_t checkpoint) {
//DUMMY_CODE(n, isn, checkpoint);
//n和checkpoint之间的最小步数
int32_t offset = n - wrap(checkpoint,isn);
//将步数加到checkpoint上
int64_t ret = checkpoint + offset;
//反着走：加2^32
return ret < 0 ? ret + (1ul << 32) : ret;
}
```

3.2 Implementing the TCP receiver 实现 TCP 接收器

3.2.1 segment_received() 接收报文

思路 & 核心代码：

- 主要针对 SYN 报文 & FIN 报文，更新 `abs_seqno`
- 写入 payload
  - 将任何数据或流结束标记推到 `StreamReassembler`，序号使用 `stream_index`
  - 设置新的 `checkpoint`

Code:

```
void TCPReceiver::segment_received(const TCPSegment &seg) {
//DUMMY_CODE(seg);

//Process syn flag
if(seg.header().syn)
{
    if(!_syn_received)
    {
        _syn_received = 1;
        _isn = seg.header().seqno;
    }
}

//Process fin flag
bool eof = 0;
if(seg.header().fin)
{
        eof = 1;
}

//Push payload
if(ackno().has_value() && !_reassembler.stream_out().input_ended())
{
        //relative seqno to stream index
        uint64_t stream_index = unwrap(seg.header().seqno, _isn, _checkpoint);
        //stream index should be in the window
        uint64_t abs_ackno = unwrap(ackno().value(), _isn, _checkpoint);
        if(stream_index + seg.length_in_sequence_space() <= abs_ackno || stream_index >= abs_ackno + window_size())
                return;

        if(!seg.header().syn)
                stream_index = stream_index - 1; //ignore syn flag

        _reassembler.push_substring(seg.payload().copy(), stream_index, eof);
        _checkpoint = _reassembler.stream_out().bytes_written();
}
}
```

3.2.2 ackno()

思路 & 核心代码：

- 用于通知对端当前接收的字节流进度
- 返回接收方未知的第一个字节的序列号
    - 将任何数据或流结束标记推到 `StreamReassembler`，序号使用 `stream_index`
    - 设置新的 `checkpoint`

Code:

```
optional<WrappingInt32> TCPReceiver::ackno() const
{
    // next_write + 1: syn flag will not push in stream
    size_t next_write = _reassembler.stream_out().bytes_written() + 1;
    next_write = _reassembler.stream_out().input_ended() ? next_write + 1 : next_write;
    return !_syn_received ? optional<WrappingInt32>() : wrap(next_write, _isn);
}
```

3.2.3 window_size()

思路 & 核心代码：

- 用于通知对端当前可以接收的字节流大小
- 返回 “first unassembled” 索引（对应于ackno的索引）和 “first unacceptable” 索引之间的距离

Code:

```
size_t TCPReceiver::window_size() const
{
    return _reassembler.stream_out().remaining_capacity();
}
```

2. Implementation Challenges:

...

...

3. Remaining Bugs:

...

...

More details and requirements of sections above can be found in `lab2_tutorials.pdf/5.submit`