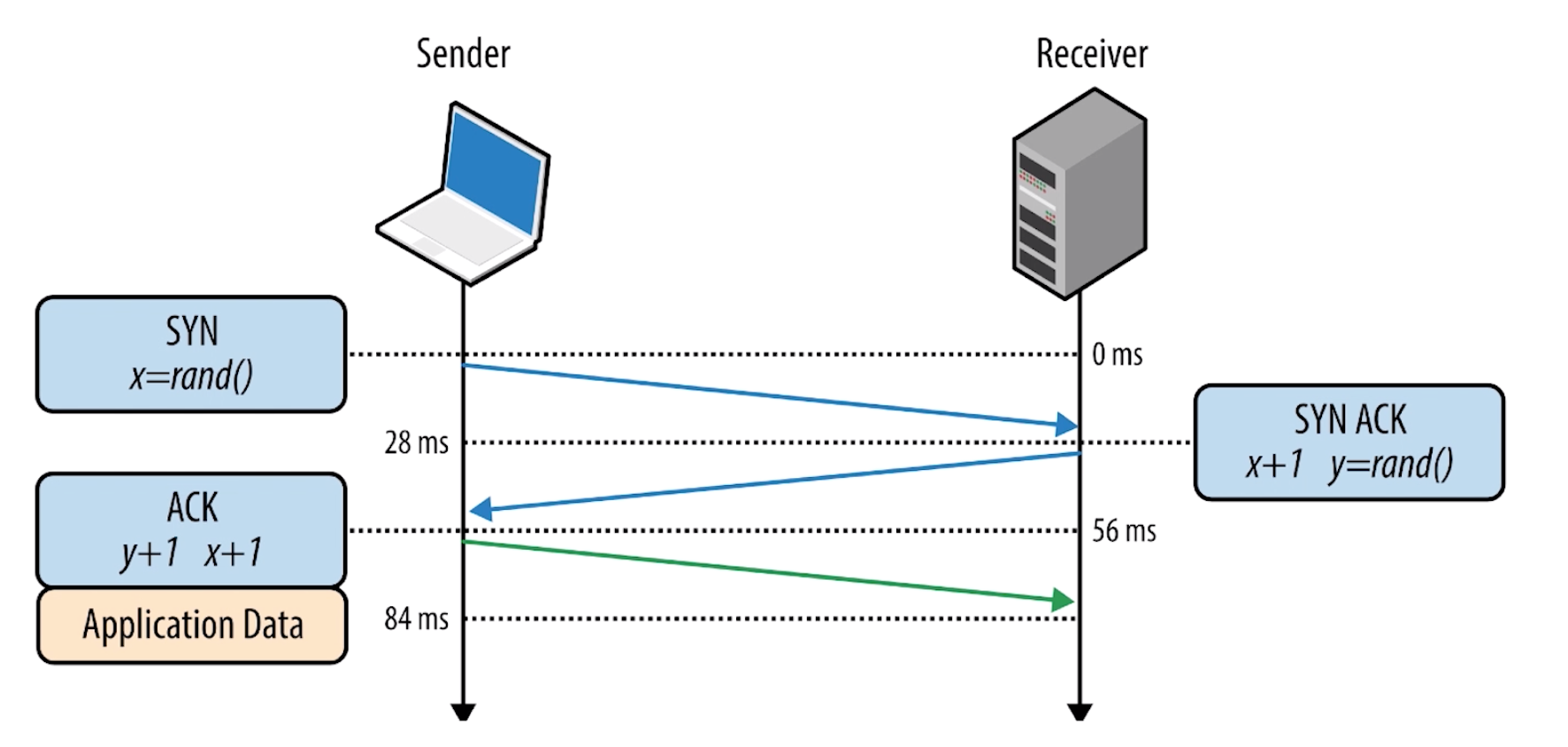
## Socket TCP 快速入门

### TCP 是什么、能做什么

### TCP 核心 API 讲解

### TCP 传输可靠性-三次握手、四次挥手

**先看三次握手的操作**，如下图所示



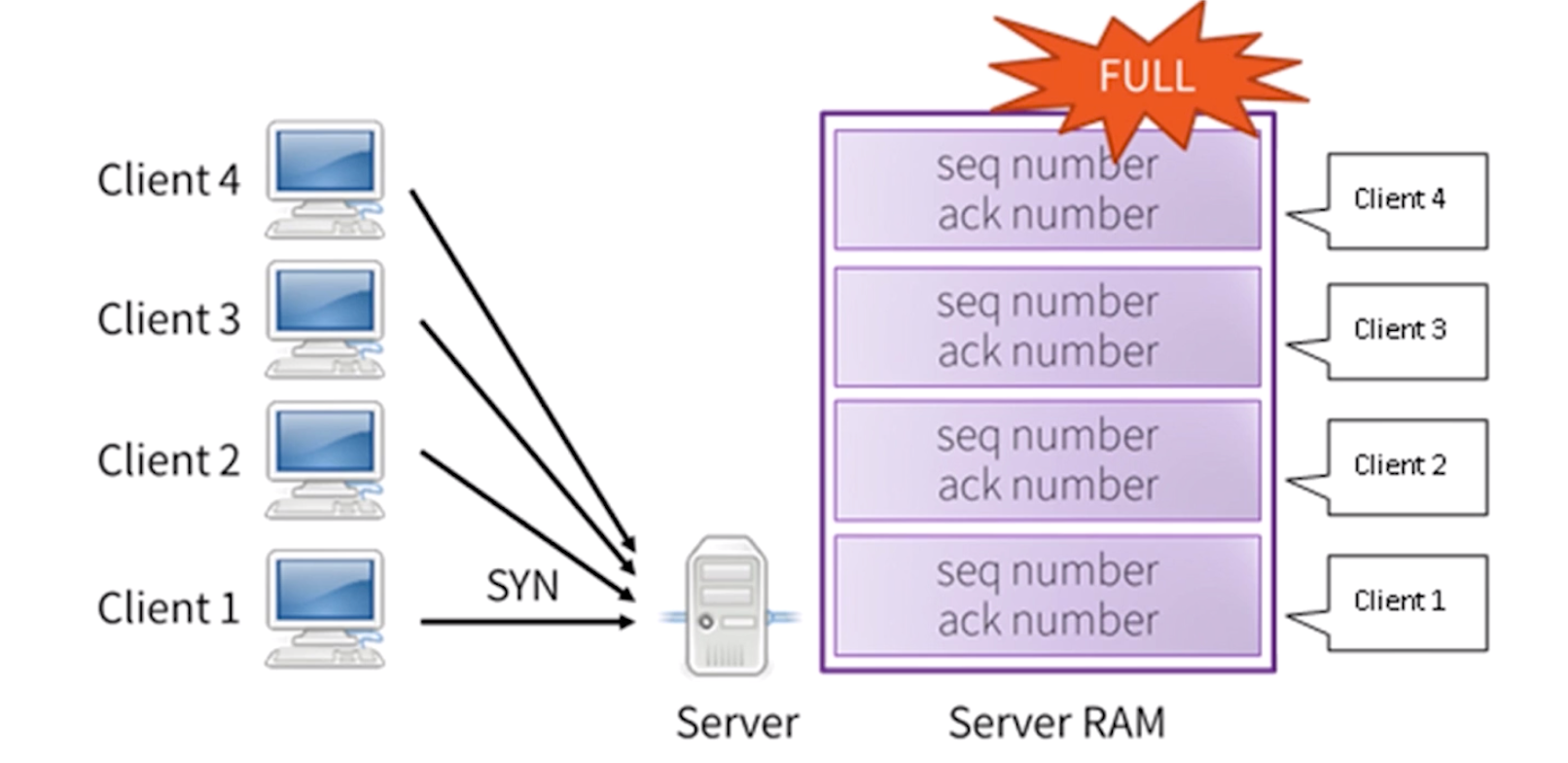
上图表明了发送方和接收方的一个流程。

第一次：客户方发起一个 x=rand() 的随机值，发送给服务器。然后进行校验，如果这个数据正确，然后组装 ack 命令。

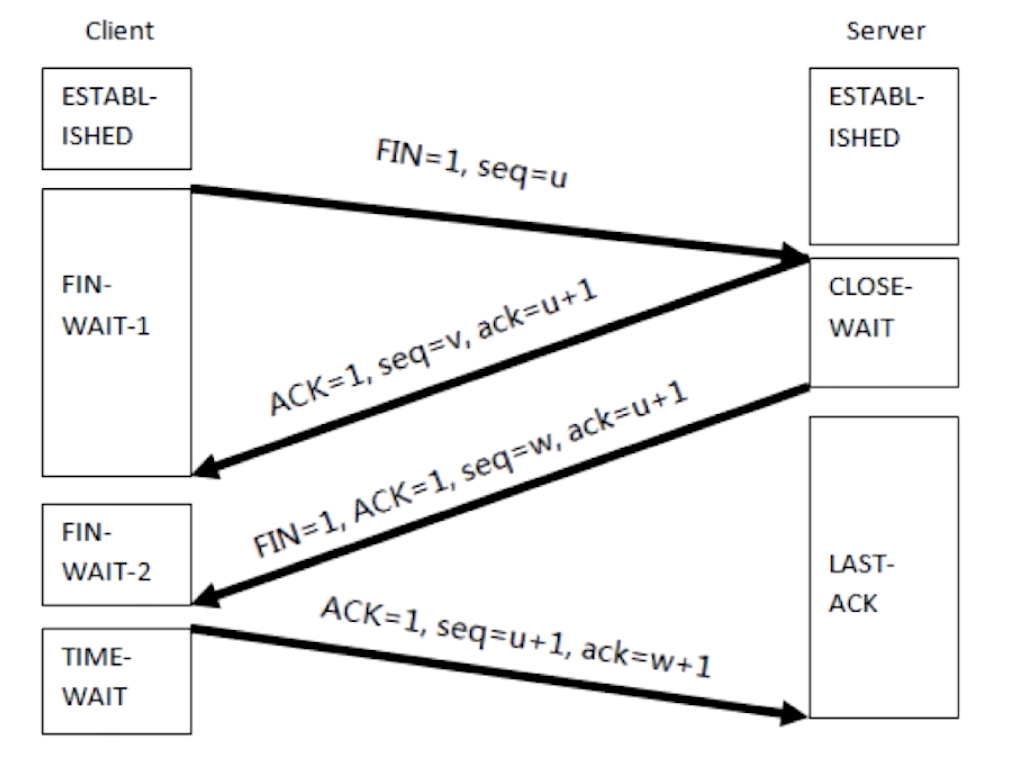
第二次：ack 命令中，对 x 进行+1操作，然后 y=rand() 操作数据返回给客户端。

第三次：收到服务方的 ack 命令后，如果收到的是 x+1 的数据，就说收到了，然后将 y+1操作。这就表示真正的连接上了。

**下面是三次握手中，数据随机的必要性**。随机数可以确定是哪个客户端，以便于客户端和服务器是一对一的正确性。



**下面说四次挥手**，如下图：



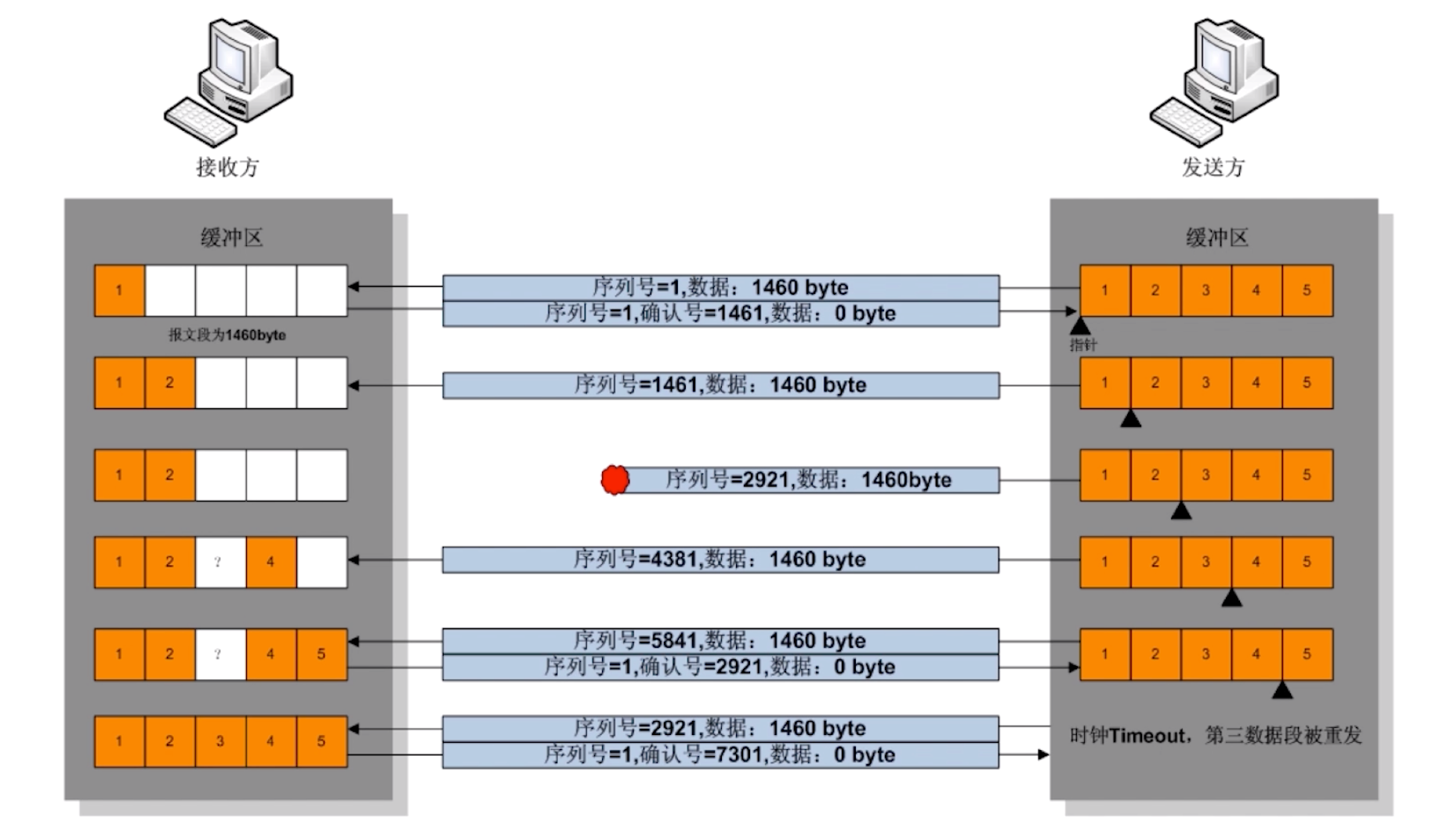
四次挥手保证了全双工的一个连接。首先断开输出流，但是输入流还是可以的。此时处于半双工状态，然后服务器发送客户端，说断开了，然后发送消息说可以断开了，这个时候就断开了输入流。此时半双工也断开。最后全双工的都断开了。

三次挥手是不确定，五次挥手是多余。

### TCP 传输的可靠性-排序、丢弃、重发

1. 排序、顺序发送、顺序组装
2. 丢弃、超时
3. 重发机制-定时器

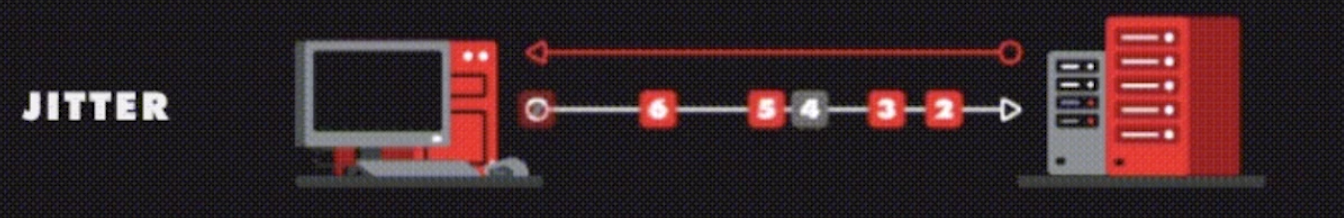
下图演示了数据发送的一个流程：



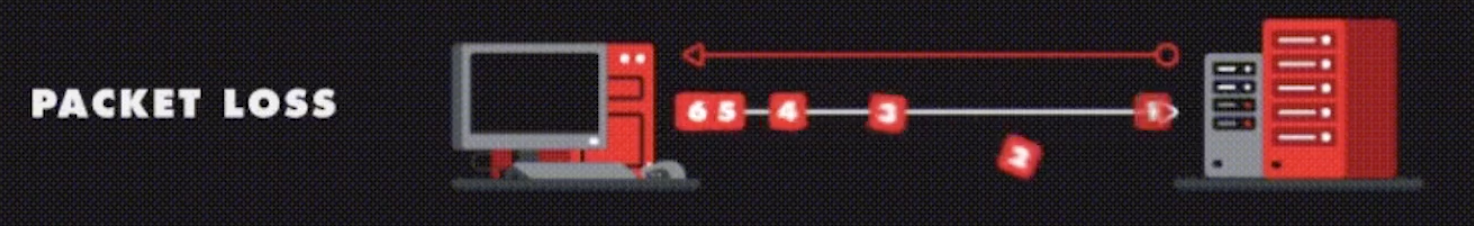
可以看见发送方有 5 个数据包。

下面看可靠性的示意图：

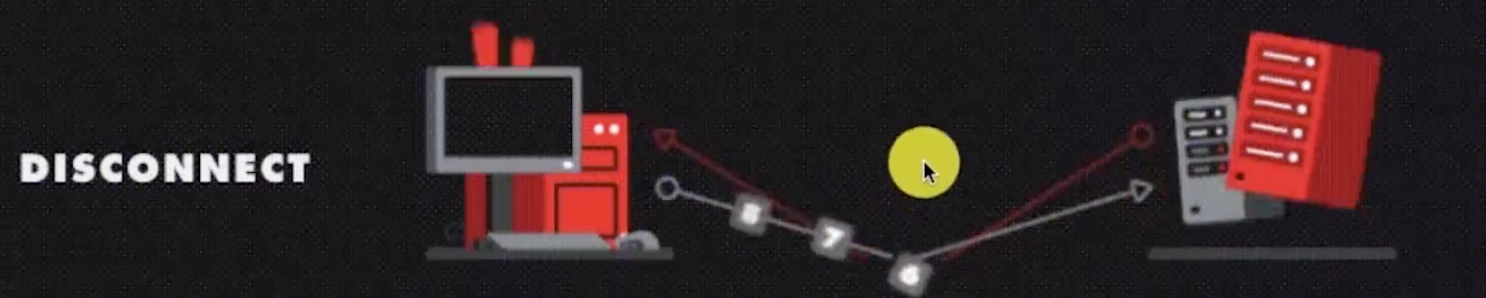
这个是排序组装的一个过程，到大服务器的顺序是可能不一样的。



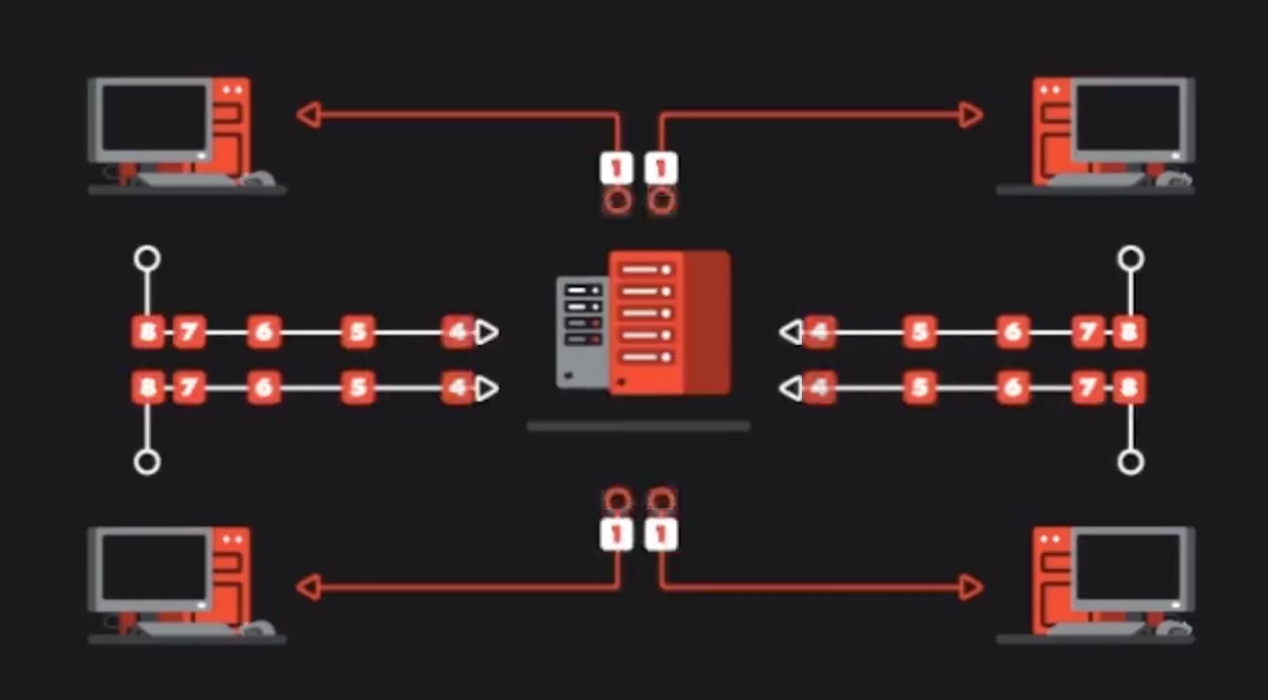
下面数据包丢失后，会进行重新发送。



如果出现断开的情况，就会进行重试进行连接，然后再发送数据。



可靠性示意图服务器一对多的情况：



### 案例实操

所有的代码都试基于第二章节（TCP 的最简单的 Demo） 改造过来的。

本节的代码不带着敲了，都是先写好了，直接讲代码

\*\*\*\*\*\*\* TCP 传输初始化配置 \*\*\*\*\*\*\*\*

1. 初始化服务器 TCP 链接监听
2. 初始化客户端发起连接操作
3. 服务器 Socket 连接处理

\*\*\*\*\*\*\* TCP基础数据传输 \*\*\*\*\*\*\*

1. 客户端发送简单字节
2. 服务器接收客户端发送的数据
3. 服务器回送消息，客户端识别回送消息