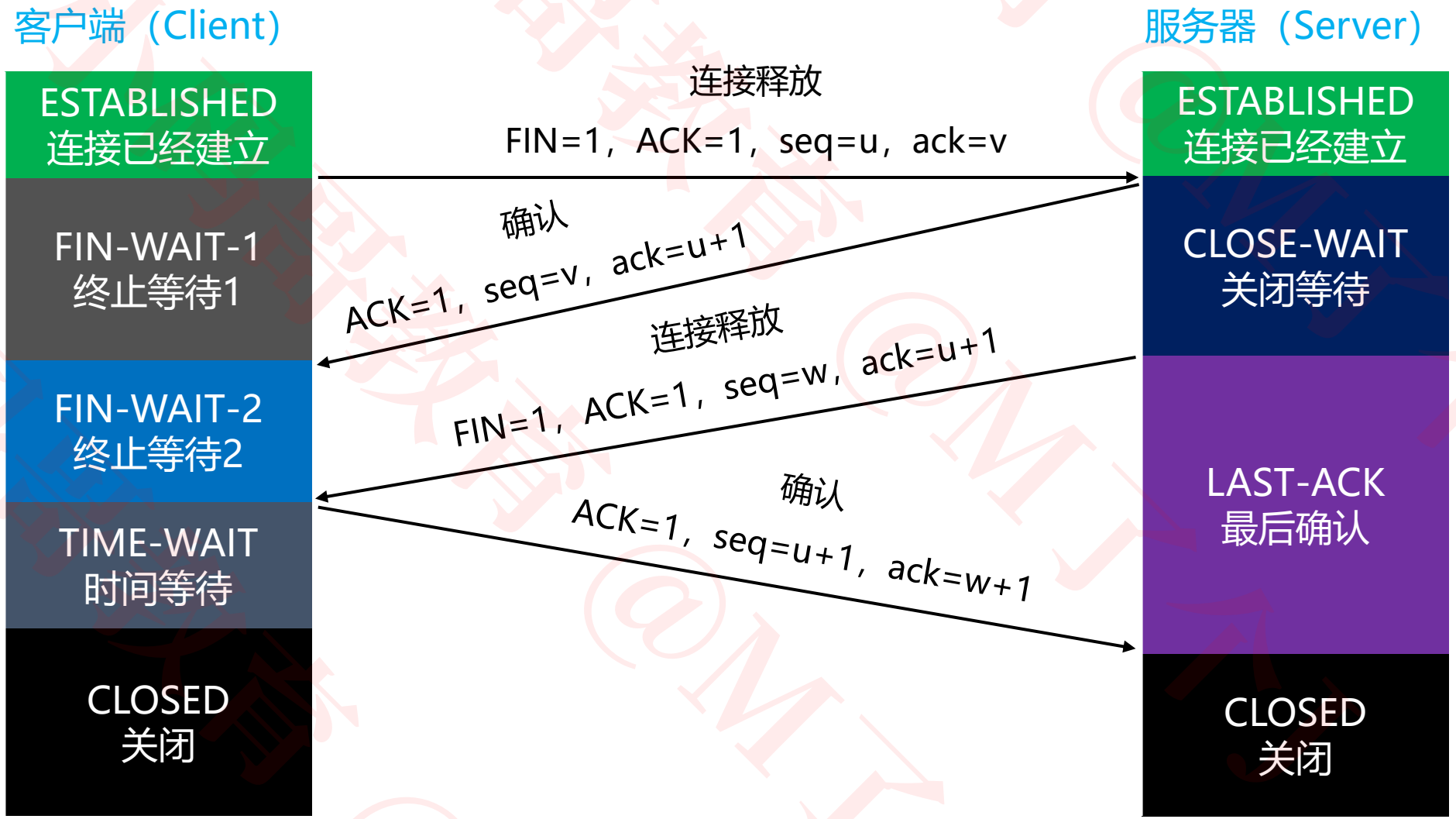
TCP – 释放连接 – 4次挥手



TCP – 释放连接 – 状态解读

- FIN-WAIT-1：表示想主动关闭连接
  - 向对方发送了FIN报文，此时进入到FIN-WAIT-1状态

- CLOSE-WAIT：表示在等待关闭
  - 当对方发送FIN给自己，自己会回应一个ACK报文给对方，此时则进入到CLOSE-WAIT状态
  - 在此状态下，需要考虑自己是否还有数据要发送给对方，如果没有，发送FIN报文给对方

- FIN-WAIT-2：只要对方发送ACK确认后，主动方就会处于FIN-WAIT-2状态，然后等待对方发送FIN报文

- CLOSING：一种比较罕见的例外状态
  - 表示你发送FIN报文后，并没有收到对方的ACK报文，反而却也收到了对方的FIN报文
  - 如果双方几乎在同时准备关闭连接的话，那么就出现了双方同时发送FIN报文的情况，也即会出现CLOSING状态
  - 表示双方都正在关闭连接

TCP – 释放连接 – 状态解读

- LAST-ACK：被动关闭一方在发送FIN报文后，最后等待对方的ACK报文
  - 当收到ACK报文后，即可进入CLOSED状态了

- TIME-WAIT：表示收到了对方的FIN报文，并发送出了ACK报文，就等2MSL后即可进入CLOSED状态了
  - 如果FIN-WAIT-1状态下，收到了对方同时带FIN标志和ACK标志的报文时
    - ✓可以直接进入到TIME-WAIT状态，而无须经过FIN-WAIT-2状态

- CLOSED：关闭状态

- 由于有些状态的时间比较短暂，所以很难用netstat命令看到，比如SYN-RCVD、FIN-WAIT-1等

TCP – 释放连接 – 细节

- TCP/IP协议栈在设计上，允许任何一方先发起断开请求。这里演示的是client主动要求断开

- client发送ACK后，需要有个TIME-WAIT阶段，等待一段时间后，再真正关闭连接
  - 一般是等待2倍的MSL（Maximum Segment Lifetime，最大分段生存期）
    - ✓ MSL是TCP报文在Internet上的最长生存时间
    - ✓ 每个具体的TCP实现都必须选择一个确定的MSL值，RFC 1122建议是2分钟

- 如果client发送ACK后马上释放了，然后又因为网络原因，server没有收到client的ACK，server就会重发FIN
  - 这时可能出现的情况是
    - ① client没有任何响应，服务器那边会干等，甚至多次重发FIN，浪费资源
    - ② client有个新的应用程序刚好分配了同一个端口号，新的应用程序收到FIN后马上开始执行断开连接的操作，本来它可能是想跟server建立连接的

TCP – 释放连接 – 疑问

- 为什么释放连接的时候，要进行4次挥手？
- TCP是全双工模式
- 第1次挥手：当主机1发出FIN报文段时
  - ✓ 表示主机1告诉主机2，主机1已经没有数据要发送了，但是，此时主机1还是可以接受来自主机2的数据
- 第2次挥手：当主机2返回ACK报文段时
  - ✓ 表示主机2已经知道主机1没有数据发送了，但是主机2还是可以发送数据到主机1的
- 第3次挥手：当主机2也发送了FIN报文段时
  - ✓ 表示主机2告诉主机1，主机2已经没有数据要发送了
- 第4次挥手：当主机1返回ACK报文段时
  - ✓ 表示主机1已经知道主机2没有数据发送了。随后正式断开整个TCP连接

TCP – 释放连接 – 抓包

- 有时候在使用抓包工具的时候，有可能只会看到“3次“挥手
  - 这其实是将第2、3次挥手合并了

```
192.168.3.3       113.96.140.220    TCP      54 6512 → 80 [FIN, ACK] Seq=661 Ack=9240 Win=262400 Len=0
113.96.140.220    192.168.3.3       TCP      60 80 → 6512 [FIN, ACK] Seq=9240 Ack=662 Win=11008 Len=0
192.168.3.3       113.96.140.220    TCP      54 6512 → 80 [ACK] Seq=662 Ack=9241 Win=262400 Len=0
```

- 当server接收到client的FIN时，如果server后面也没有数据要发送给client了
  - 这时，server就可以将第2、3次挥手合并，同时告诉client两件事
  - ✓ 已经知道client没有数据要发
  - ✓ server已经没有数据要发了