面试常问之TCP与UDP

味道_3a01 (/u/ca36c866259e) + 关注
0.1 2018.08.29 16:13* 字数 3056 阅读 64 评论 0 喜欢 1
(/u/ca36c866259e)

TCP: 传输控制协议（英语：Transmission Control Protocol，缩写为TCP）是一种面向连接的、可靠的、基于字节流的传输层通信协议，由IETF的RFC 793定义。

UDP:用户数据报协议（英语：User Datagram Protocol，缩写为UDP），又称使用者资料包协定，是一个简单的面向数据报的传输层协议，正式规范为RFC 768。

共同点

- 传输层协议

不同点

- UDP 继承 IP 的特性，不保证不丢失，不保证按顺序到达。TCP 就是提供可靠的数据，无差错、不丢失、不重复、按循序到达。

- TCP提供有保证的数据传输，而UDP不提供

> 这意味着TCP有一个特殊的机制来确保数据安全的不出错的从一个端点传到另一个端点，而UDP不提供任何这样的保证。

- TCP是面向链接的，虽然说网络的不安全不稳定特性决定了多少次握手都不能保证连接的可靠性， 但TCP的三次握手在最低限度上（实际上也很大程度上保证了）保证了连接的可靠性；

- UDP不是面向连接的，UDP传送数据前并不与对方建立连接，对接收到的数据也不发送确认信号， 发送端不知道数据是否会正确接收，当然也不用重发，所以说UDP是无连接的、不可靠的一种数据传输协议

> 也正由于上面的特点，使得UDP的开销更小数据传输速率更高，因为不必进行收发数据的确认，所以UDP的实时性更好。
>
> 所以采用TCP传输协议的MSN比采用UDP的QQ传输文件慢
>
> 但并不能说QQ的通信是不安全的，因为程序员可以手动对UDP的数据收发进行验证， 比如发送方对每个数据包进行编号然后由接收方进行验证啊什么的

- UDP 是数据格式基于数据报，一个一个的发，一个一个的收。TCP 是面向字节流，发送的是一个流数据。

- UDP 是不处理堵塞，应用需要发，就会发送。TCP 还拥有堵塞控制，TCP 会根据网络环境调整发包的频率。

- UDP 支持多播和广播

TCP、UDP对比

特性	TCP	UDP

特性	TCP	UDP
可靠性	可靠	不可靠
连接性	面向连接	无连接
报文	面向字节流	面向报文
效率	传输效率低	传输效率高
双工性	全双工	一对一、一对多、多对一、多对多
流量控制	有(滑动窗口)	无
拥塞控制	有(慢开始、拥塞避免、快重传、快恢复)	无

TCP和UDP协议的一些应用

应用层协议	应用	传输层协议
SMTP	电子邮件	TCP
TELNET	远程终端接入	
HTTP	万维网	
FTP	文件传输	
DNS	名字转换	UDP
TFTP	文件传输	
RIP	路由选择协议	
BOOTP，DHCP	IP 地址配置	
SNMP	网络管理	
NFS	远程文件服务器	
专用协议	IP 电话	
专用协议	流式多媒体通信	

image

引申

TCP三次握手

TCP 是面向连接的，数据通信之前必须要建立连接，这个连接可以是 1 对多。

TCP 三次握手的过程

image

上图展示三次握手的流程图，文字描述过程如下： 客户端和服务端从 CLOSED 的状态变成在线状态。服务端需要处于 LISTEN. 1、客户端向服务端发起一个连接。报文中包含了： SYN = 1，「序列号」的值， 发送之后客服端进入到 SYN-SENT 状态。

2、服务端接收到 SYN 和 序列号信息后，需要对这个报文进行的确认。服务端收到客户端发送过来的「序列号」的值，服务端会返回一个「确认号」的值（序列号+1）。同时返回 SYN = 1 表示还在同步阶段。 ACK = 1 表示确认号有意义。 序列号：表示服务端的序列号 y。 确认号：针对客户端发起连接的一个确认号。 服务端发送之后，进入到 SYN-RCVD 状态。

3、客户端接收到服务端返回的数据，使得客户端这一边的相关通信建立，并且同步了相关序列号和确认号的数值，但是服务端还没有接收到客户端回复的确认号。所以客户端需要再发送一个数据到服务端，主要包含： ACK = 1，这是确认号的数据有意义。 SYN = 0 序列号已经同步完成，不需要再同步序列号。 序列号：因为 SYN 已经不是第一次同步序列号的信息了。这个时候的序列号，就表示的是一个单纯的基于序列号最新的数据包的序列号。其值为：x+1。 确认号：返回服务端序列号 y 的确认号，为 y+1。 发送这个值之后，客户端进入到 ESTABLISHED 状态，服务端接受到这个值之后也进入到 ESTABLISHED 状态。然后就可以开始传递数据通信了。

至此**三次握手**已经完成。

通俗易懂的描述 (https://www.jianshu.com/p/b0082662b22e)

TCP 为什么进行是三次握手?

为什么 TCP 需要进行三次握手，这个问题很多人的解释都引用了

> 为了防止已失效的连接请求报文段突然又传送到了服务端，因而产生错误。——谢希仁的《计算机网络》

首先我们需要明白三次握手的意义在于什么，也就是说三次握手需要达到的一个目的是什么， 简单来说，其实是为了**同步序列号和确认号的相关信息**，而序列号和确认号又是保障数据的正确性以及窗口滑动机制的最基本的根据。所以我最需要明白的可能还是为什么要同步序列号就是为什么要三次握手：（举个例子来说明） 如果 A 给 B发送一个数据包，这个包由于网络的原因，很久才到 B 端，而这段时间，A 和 B 已经断开，并且重新建立了连接关系。没有相关同步序列号和确认号，B 端认为这个数据还是认为 A 这个时候要发给我的，但是其实这个数据上一次传输的数据，相当于这次传输的数据中插入了其他的数据，那么就会导致这次的数据出现异常。

为什么不是两次

对于 A 来看说，两次实现了信息一来一回，但是 TCP 的数据的可靠性，如果要实现对于 B 来说的信息的一来一回，那么第三次是一定要存在的。**对两端的数据一来一回才是建立可靠连接的基本要求。**

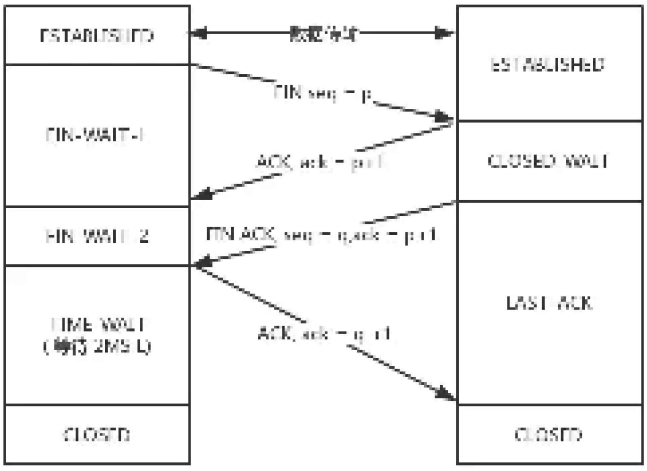
为什么不是四次

理论上其实这个同步序列号和确认号的过程，大于三次也不是没有的，应该说几十次上百次都是可以的，但是三次握手的过程已经实现了序号数据同步，在进行太多次的序号同步，已经没有意义。无故浪费宽带。

TCP 四次挥手

当通信完成，当然需要能够进行连接断开。

TCP 四层挥手的过程：



image

1、任意一段（ A 端）发出一条 FIN 的数据。数据包含： FIN = 1, 序列号 A 端发出这个数据之后，进入到 FIN - WAIT -1 的状态，等待 B 端的回复.

2、B 收到 A 端的 FIN 的信息就回复一个数据，表示已经知道 A 端请求断开连接这个事情了。此时 B 端进入到 CLOSED - WAIT 的状态。A 端接收到这个数据进入到 FIN - WAIT -2 的状态。 数据中包含： ACK = 1 确认号

3、B 端给 A 端发起一次 FIN 的数据。请求结束连接，发送完成之后 B 端进入到 LAST - ACK 状态。发送的数据包含： FIN = 1; 序列号 ACK = 1; 确认号

4、 A 端接收到这个 B 端发送的 FIN 数据进入到 TIME- WAIT 状态，同时回复 B 端，已经接收到了 FIN 数据。回复的包中包含： ACK = 1; 确认号

为什么需要设置一个 TIME-WAIT 的时间

这个文章讲的特别好 (https://blog.csdn.net/smileiam/article/details/78226816)

> 一、保证TCP协议的全双工连接能够可靠关闭 先说第一点，如果Client直接CLOSED了，那么由于IP协议的不可靠性或者是其它网络原因，导致Server没有收到Client最后回复的ACK。那么Server就会在超时之后继续发送FIN，此时由于Client已经CLOSED了，就找不到与重发的FIN对应的连接，最后Server就会收到RST而不是ACK，Server就会以为是连接错误把问题报告给高层。这样的情况虽然不会造成数据丢失，但是却导致TCP协议不符合可靠连接的要求。所以，Client不是直接进入CLOSED，而是要保持TIME_WAIT，当再次收到FIN的时候，能够保证对方收到ACK，最后正确的关闭连接。

> 二、保证这次连接的重复数据段从网络中消失 再说第二点，如果Client直接CLOSED，然后又再向Server发起一个新连接，我们不能保证这个新连接与刚关闭的连接的端口号是不同的。也就是说有可能新连接和老连接的端口号是相同的。一般来说不会发生什么问题，但是还是有特殊情况出现：假设新连接和已经关闭的老连接端口号是一样的，如果前一次连接的某些数据仍然滞留在网络中，这些延迟数据在建立新连接之后才到达Server，由于新连接和老连接的端口号是一样的，又因为TCP协议判断不同连接的依据是socket pair，于是，TCP协议就认为那个延迟的数据是属于新连接的，这样就和真正的新连接的数据包发生混淆了。所以TCP连接还要在TIME_WAIT状态等待2倍MSL，这样可以保证本次连接的所有数据都从网络中消失。

TCP 为什么进行是四次挥手?

断开比连接更复杂，比较直接的理解是资源回收比资源分配会更麻烦。使得所有资源能够有效并且不产生错误的情况下释放。

三次握手的本质是：将“四次握手”中的第二次、第三次握手合为一次，因为“四次握手”中的第二次、第三次握手都是由B向A传递报文，而且这两次发送报文的目的允许这两次报文合并为一次。 那么，TCP四次挥手中的第二次、第三次挥手，能否也能合为一次呢?

答案是否定的。将TCP四次挥手中的第二次、第三次挥手，合为一次。也就是将CLOSE_WAIT状态的停留时间变为0。然而，B之所以存在CLOSE_WAIT状态，是因为B可能还存在着需要发送给A但是未发送的数据，如果存在着这些数据，那么这个状态的时间，就是用来发送这些数据的，所以，TCP四次挥手中的第二次、第三次挥手无法合并为一次。所以，也就无法实现“TCP三次挥手”。

参考

TCP、UDP、Socket、HTTP你不知道的故事 (https://www.jianshu.com/p/74de8df5f686)
UDP 的模样和你想的是否一样 (https://www.jianshu.com/p/ed33413c1e3b)
跟着动画学习 TCP 三次握手和四次挥手 (https://www.jianshu.com/p/b0082662b22e)
TCP、UDP以及TCP滑窗，它们的区别 (https://www.jianshu.com/p/742897d6b0f8)

小礼物走一走，来简书关注我

赞赏支持

日记本 (/nb/23730690)

举报文章 © 著作权归作者所有

味道_3a01 (/u/ca36c866259e)
写了 33466 字，被 3 人关注，获得了 14 个喜欢
(/u/ca36c866259e)

+关注

spring热爱者

喜欢 1

更多分享