1. 传输层
2. 概述

TCP/IP中有两个代表性的传输层协议TCP和UDP，TCP提供可靠的通信传输，而UDP则用于将广播和细节控制交给应用的通信传输。

1. 通信处理

TCP/IP中的多数应用协议多以客户端/服务端的形式运行，这些服务端程序在UNIX系统中称为守护进程，如HTTP的服务端程序是httpd，而ssh的守护进程是sshd。在UNIX中不需要将这些守护进程逐个启动，而是启动一个代表它们接受客户端请求的inetd（互联网守护进程）即可，它是一种超级守护进程，收到客户端请求后会复制新的进程并转换为各个守护进程。通过端口号来识别请求时发给哪个守护进程的。

1. TCP和UDP
2. TCP

TCP是面向连接的，可靠的流协议，流是不间断的数据结构，具有流控制，拥塞控制等功能。

1. UDP

UDP是不具有可靠性的数据报协议，细微的处理会交给上层的应用来完成，不能保证消息一定会到达。

1. 使用

TCP为应用提供可靠传输，UDP适合高速传输以及实时性高的通信或广播通信。

1. 端口号

TCP/IP通信通常采用5个信息来识别一个通信，分别是源地址，目标IP地址，协议号，源端口号，目标端口号，只要一项不同，就会被认为是其他通信。

1. 端口号的确定

有些端口号是固定的，用于知名应用，如HTTP是80，FTP是21等等，知名端口号一般是0-1023分配，我们应该避免分配这些端口。

1. 端口号与协议

端口号由其使用的传输层协议决定，不同的传输协议可以使用相同的端口号。数据到达IP层后，会先检查IP首部的协议号，如果是TCP则传给TCP模块，如果是UDP则传给UDP模块。

1. TCP与UDP
2. TCP
3. 概述

TCP实现了数据传输中的各种控制功能，可以进行丢包的重复控制，对次序乱掉的分包进行顺序控制。TCP作为面向有连接的协议，只有在确认通信对端存在时才会发送数据，因此在IP这种无连接的网络上也能够实现高可靠性通信。

1. 序列号与确认应答

TCP中，当发送端的数据到达接受主机时，接收端主机会返回一个已收到消息的通知，这个消息叫做 确认应答（ACK），否定确认应答是（NACK），发送端将数据发出后会等待对端的ACK，如果有应答说明数据已经成功抵达对端。如果在一定时间内没有等到ACK，发送端就会认为数据已经丢失，进行重发，但是这并不意味着数据一定丢失，有可能是对端已经收到，但是ACK丢失了，这样会导致接收端会收到重复的消息，因此它就需要放弃这些重复的消息。

这些应答控制都可以通过序列号实现，序列号是按照顺序给发送数据的每一个字节都标上号码的编号，接收端查询接受数据TCP首部中的起始序列号和数据的长度，将自己下一步应该接受的序列号作为ACK返回给发送端，这样就可以实现可靠传输。

1. 重发超时如何确定

重发超时是指在重发数据之前，等待ACK到来的哪个特定时间间隔，最理想的是找到一个最小时间，能够保证ACK一定能在这个时间返回。TCP在每次发包时都会计算往返时间（RTT）和偏差，将RTT和偏差相加就差不多是重发时间了。

UNIX和WINDOWS中，超时都以0.5秒为单位进行控制，重发次数也是有限制的，超过一定次数，则会强制关闭连接，通知应用好吃呢关系通信异常终止。

1. 连接管理

TCP面向有连接是指在通信开始前，做好通信两端的准备工作。首先通过TCP首部发送一个SYN包作为建立连接的请求等待ACK，如果对端发来ACK，则认为可以通信，否则就不会通信。此外在通信结束时会进行断开连接的处理（FIN包）。

三次握手指的是建立TCP连接的过程，发送端发送SYN，接收端返回针对SYN的ACK，发送端收到ACK，返回针对该ACK的ACK，这就是三次请求过程。而断开连接时，发送端发送FIN请求断开，接收端发送针对FIN的ACK以及FIN给发送端，最后发送端发送针对该FIN的ACK。

一个连接的建立与断开过程至少需要来回发送7个包才能够完成。

1. TCP以段为单位发送数据

在建立TCP连接的时候，会确定发送数据包的单位，称为最大消息长度MSS，最理想的情况是IP中不会被分片处理的最大数据长度。TCP在传送大量数据时，是以MSS的大小对数据进行分割发送的，MSS是在三次握手时，在两端主机之间计算得出的。两端都会在TCP首部写入自己认为合适的MSS，然后取较小的一个使用。

1. 利用窗口提高速度

如果TCP仅仅以一个段为单位，每发送一个段进行一个ACK处理，那么效率很低。为了提高效率，TCP引入了窗口概念，ACK以更大的窗口为单位进行，发送端主机在发送了一个段之后不用一直等待ACK，而是继续发送。窗口大小就是指无需等待ACK而可以继续发送数据的最大值。

这个机制通过使用大量的缓冲区，实现了对多个段同时进行ACK的功能，如每个段是1000字节，设置窗口为4000字节，那么发送端可以连续发送4000字节，并且等待ACK，如果第一个段1-1000的ACK返回了，则窗口向前移动一段，继续发送4001-5000的段，如果出现了数据丢包现象，发送端要负责重传，缓存需要将窗口中的包缓存起来，用于重传。在滑动窗口以外的部分包括尚未发送的以及已经收到ACK的就不需要缓存了。这种机制称为滑动窗口控制。

1. 窗口控制与重发控制

在使用窗口控制后，如果接收端已经收到了报文段，但是丢失了ACK，也不需要重发了，因为它会收到下一个段的ACK，这个ACK代表着前面的都已经收到了。但是如果是丢失了报文段，则发送端会一直收到前一个报文段的ACK，通知发送端报文丢失，如果收到三次，则发送端会对该报文段进行重发处理，这种机制比超时管理更加高效，因此成为高速重发控制。

1. 流控制

TCP可以让发送端根据接收端的实际接收能力控制发送的数据量，这就是流控制，接受主机将自己可以接受的缓冲区大小放入TCP 首部的某一字段中通知发送端，这个值越大说明网络的吞吐量越高。如果接收端的这个缓冲区一旦溢出，则该窗口的值会变小，从而控制发送量。如果这个缓冲区窗口的通知更新在中途丢失，则发送端会时不时地发送一个叫做窗口探测的数据段，此数据段仅含一个字节，以获取最新的缓冲区窗口大小。

1. 拥塞控制

如果在通信刚开始就发送大量数据，可能会导致很大问题，因此TCP通过慢启动来解决，定义了一个拥塞窗口，在慢启动时设置为1个数据段的大小，之后没收到一次ACK就将窗口大小增大1，这样吞吐量会慢慢增加，不至于直接冲垮网络。

但是实际的TCP没有设置慢启动阈值，拥塞窗口会慢慢增大，在超时重发时，会设置为当前拥塞窗口的一半大小。

1. 延迟ACK

接收端如果每次都立即回复ACK的话，可能会回复一个较小的窗口，因为刚接受完数据，缓冲区已满，当发送端收到这个小窗口的通知后，会以它为上限发送数据，从而降低了网络的利用率。因此，采用延迟ACK的方式，尽量地扩大窗口大小，提高利用率。

1. 捎带ACK

TCP的ACK和回执数据通过一个包发送，这种称为捎带ACK。有些应用发送出去的消息到达对端后，经过处理会返回一个回执，这种时候使用捎带ACK，如电子邮件的SMTP和POP，远程登录中的回送校验，即从键盘输入的字符到达服务器以后再返回来显示给客户端，这些都是需要对端进行处理的情况，因此ACK需要等待处理完成再一起发送回来，这就需要开启延迟ACK功能了。

1. UDP

UDP是用户数据报的缩写，不提供复杂的控制机制，利用IP提供面向无连接的通信服务，传输途中出现丢包UDP不会负责重发，包到达顺序乱掉也不会进行纠正。如果需要这些细节控制，则需要交由上层的采用UDP的应用程序来处理。它适用于包总量较少的通信，视频，音频通信，广播通信。