第三层 运输层

2020年10月30日

11:29

运输层协议：为运行在不同主机上的**应用进程**提供逻辑通信。

网络层协议：为**主机之间**提供逻辑通信。

TCP服务：

1）数据交付

2）差错检查

3）可靠数据传输

4）拥塞控制

UDP服务：

1）数据交付

2）差错检查

TCP与UDP最基本责任：将两个系统间IP的交付服务扩展，为运行在端系统上的俩个进程之间交付服务。

交付：

1）多路交付

2）多路分解

UDP（无连接）

优点：

1.关于发送什么数据以及何时发送应用层控制更为精细。

2.无需连接建立。

3.无连接状态

4.分组首部开销小

UDP报文段：

|  |  |
| --- | --- |
| 源端口号 | 目的端口号 |
| 长度 | 效验和 |
| 应用数据报文（报文） |  |

效验和目的：使接收方检测到何时出现了错误。

**TCP**

流水线差错恢复：

1）回退N步 GBN (又称滑动窗口协议)

2）选择重传 SR

回退N步协议：允许发送方送多个分组（当有多个分组可用时）而不需要等待确认，但它也受限于在流水线中未确认的分组数不能超过某个最大允许数N。

GBN协议中，接受方丢弃所有失序分组。

优点：接受缓存简单，接受方不需要缓存任何失序分组。

缺点：随后对该分组的重传也许会丢失或出错，因此甚至需要更多重传。

计算机生成了可选文字:
已确认
基序号
窗囗长度

选择重传（SR）协议：通过让发送方仅重传那些它怀疑在接受方出错（即丢失或受损）的分组而避免了不必要的重传。

计算机生成了可选文字:
已确认
基序号
里色已确认
窗囗长度

**窗口长度必须小于或等于序号空间大小的一半**

TCP面向连接（三次握手，四次挥手）

TCP双全工服务

TCP点对点 无法进行多播

TCP可从缓存中取出，放入报文段的数量受限于最大报文段（MSS）

MSS通常根据最初确定的由本地发送主机发送的最大链路层帧长度（即最大传输单元（MTU））设置。

例：以太网，PPP链路层都具有1500Byte的MTU，TCP/IP首部长度通常40Byte，所以MSS为1460Byte。

TCP：

1）首部字段

2）数据字段

计算机生成了可选文字:
序号
确认后
长度
湃端囗号
保留
未用
因特网裣输和@
目的端囗号
接牧窗囗
经济据指针@

源端口号和目的端口号确定一条TCP连接。

确认号是期望从另外一主机收到的下一字节序号。

序号是该报文段首字节的字节流编号。

**控制位**：URG、ACK、PSH、RST、SYN、FIN。

URG：紧急指针标志。

ACK：确认序号标志。

PSH：推动标注位，尽快交给程序。

RST：重置连接标志（例如：目的地址端口号更改）。

SYN：同步序号。

FIN：finish标志，用于释放连接。

接受窗口（rwnd）：滑动窗口大小

EstimatedRTT（估计往返时间）=(1-a)×EstimatedRTT+a×SampleRTT（样本时间）

a一般取0.125（1/8）

指数加权移动平均（EWMA）

RTT偏差DevRTT

DevRTT=(1-B)×DevRTT+B|SampleRTT-EstiamtedRTT|

B一般取0.25

TimeoutInterval（重传时间）=EstimatedRTT+4×DevRTT

**快速重传：**TCP收到3个冗余ACK，开始重传。

**流量控制：**

TCP提供了流量控制服务：消除发送方时接受方缓存溢出的可能性。

**发送方维护接受窗口**

当主机B的接收窗口为0时，主机A继续发送只有一个字节数据的报文段，这些报文段将会被接受方确认。最终缓存开始清空，并且确认报文段里将包含一个非零的rwnd值。

**三次握手：**

计算机生成了可选文字:
SYN}E文
=clienisn
SYNACKfi2
Seq=severisn,
ack=clientisn+1
分配缓存与变量
SYN=O
分配TCP缓存和变量

**四次挥手：**

计算机生成了可选文字:
FIN
AC
关闭成功
FIN
ACK
寸间一般为305，1分钟
定时等待

**拥塞控制：**

方法：

1.端到端（TCP使用）

2.网络辅助

TCP采用方法：让每一个发送方根据所感知的网络拥塞程度来限制其能向连接发送流量的速率。

运行在发送方的TCP拥塞控制机制跟踪一个额外变量，即拥塞窗口。(cwnd)

丢包事件：

1）超时

2）3个冗余ACK

发送方速率≈cwnd/RTT byte/s

发送方（出现丢包事件）认为路径上出现拥塞指示。

TCP使用确认（ACK）来触发（或计时）增大它的拥塞窗口（cwnd）长度。

TCP拥塞控制算法：

1）慢启动

2）拥塞避免

3）快速恢复

**慢启动**：cwnd值以1个MSS开始并且每当传输报文，首次被确认就增加一个MSS。（指数增长）

结束事件：

1）存在一个超时指示的丢包事件，cwnd设置为1个MSS重新开始慢启动。“慢启动阈值”（ssthresh）=cwnd/2。

2）当cwnd=ssthresh，结束慢启动，开始拥塞避免。

3）检测到3个冗余ACK，慢启动结束，开始快速恢复。

**拥塞避免（线性增长）**

对于TCP发送方无论何时到达一个新的ACK，就将cwnd增加一个MSS（MSS/cwnd）字节。（每RTT 1MSS）

结束事件：

1）超时，拥塞避免结束，开始慢启动。

（ssthresh=cwnd/2,cwnd=1MSS）

2）三个冗余ACK，拥塞避免结束，开始快速恢复。

（ssthresh=cwnd/2,cwnd=ssthresh+3×MSS）

**快速恢复**：对于引起TCP进入快速恢复状态的缺失报文段，对收到的每个冗余的ACK，cwnd的值增加一个MSS。

结束事件：

1）丢失报文段ACK到达，快速恢复结束，拥塞避免开始，（cwnd=ssthresh）

2）超时，快速恢复结束，慢启动开始。（ssthresh-cwnd/2,cwnd=1MSS）

TCP拥塞控制加性增，乘性减（AIMD）