Transport Layer HW

R2

a

在发送家庭中, 写信人将信交给家庭代表, 并告知收信人的名字和家庭地址. 家庭代表在信上"贴"上收信人名字, 将信装在一个信封里, 在信封上写上家庭地址, 然后将这封信交给邮政系统.

在接收家庭中, 家庭代表从邮政系统拿到信, 取下信封, 取下信上的收件人名字, 最后将信交给收信人.

b

不需要. 邮政系统只需要根据信封上的收信人家庭地址, 把信送到正确的家庭, 并交给家庭代表即可.

R14

a

第一个报文段有 110-90, 即 20 个字节.

b

接收端发回确认号 90. 这是接收端期望收到的下一字节的编号.

P24

a

序号 409, 源端口号 1028, 目的端口号 80.

b

确认号 409, 源端口号 80, 目的端口号 1028.

c

确认号 359.

d

我不会画图, 所以用文字描述时间进程.

A 发送报文段 1, 序号 359, 50 字节.

A 发送报文段 2, 序号 409, 80 字节.

B 收到报文段 1, 发送回复报文段 3, 确认号 409.

B 收到报文段 2, 发送回复报文段 4, 确认号 489.

报文 3 丢包.

计时器超时, A 重传报文段 1 的副本 5, 序号 359, 50 字节.

A 收到报文段 4.

B 收到报文段 5, 发送回复报文段 6, 确认号 489.

A 收到报文段 6.

P34

我会先对整个流程做出解释, 再直接给出问题 a-g 的答案.

首先必须指出, 本题的图像是有问题的, 至少与书中的 TCP 拥塞控制 FSM 描述有不可调和的矛盾. 我会按照我所认为正确的图像作答, 并以书中的 FSM 为唯一标准.

在第 1 轮, 发送方进入慢启动阶段, 接下来一直没有发生丢包事件, 窗口大小每轮加倍.

到了第 6 轮, 窗口大小大于等于预先设定的阈值, 进入拥塞避免阶段. 由于第 5 轮窗口大小为 16 时没有触发阈值, 而第 6 轮窗口大小为 32 时触发了, 可以断定初始阈值在 17-32 之间 (含端点). 此后仍没有发生丢包事件, 窗口大小线性增长, 直到第 16 轮.

此时, 由于窗口大小从 43 降到 21, 可以断定发生了丢包事件. 但窗口大小没有降到 1, 说明发生的不是计时器超时, 而是检测到了重复三次的 ACK. 但是, 根据 FSM 描述, 此时应进入快速恢复阶段, 将阈值设为原先窗口大小的一半, 即 21, 并将窗口大小设置为阈值加三个 MSS, 即 24 , 而不是图像所示的 21. **这只能说明, 图像有误, 第 17 轮的窗口大小只能是 24 而非 21. 第 17-22 轮的窗口大小都必须向上平移.**

当快速恢复成功 (即收到新 ACK), 根据 FSM 描述, 应把窗口大小设为阈值, 并进入拥塞避免阶段. **图像中并没有这个"窗口大小减少到阈值"的过程, 无论是否合理, 只能认为, 发送方一直处于快速恢复阶段, 直到第 22 轮后, 计时器超时, 窗口大小降到 1, 阈值设为当前窗口大小的一半, 即 14.5 (此时窗口大小应为 29 而不是图像所示的 26), 并转到慢启动阶段.**

之后的 22-26 轮, 发送方一直处于慢启动阶段, 窗口大小每轮加倍.

a

第 1-6 轮和第 23- 26 轮.

b

第 6-16 轮.

c

重复三次的 ACK.

d

超时.

e

17 到 32 中的某一值. 考虑实际的话, 有很大可能是 32.

f

21.

g

14.5. 如果按照 C 语言标准进行舍入, 将会是 14.

h

第 7 轮.

在最初的慢启动阶段, 每收到一个新 ACK 就将窗口大小增大 1. 到第 6 轮开始前, 发送方共收到了 31 个 ACK (32 减 1), 由于一直没有发生丢包事件, 也就是共发送了 31 个报文段. 第 6 轮发送方又发送了 32 个报文段, 至此共发送了 63 个报文段. 第 7 轮发送方又发送了 33 个报文段, 第 70 个报文段就在其中.

i

窗口大小 7, 阈值 4.

在检测到这次重复三次的 ACK 时, 窗口大小为 8. 接下来, 发送方会进入快速重传阶段, 将阈值设定为原先窗口大小的一半, 即 4, 并将窗口大小设置为阈值加 3 MSS, 即 7.

P36

a

7 RTT.

每经过一个 RTT, 发送方收到一批 ACK, 并将窗口大小扩大 1 MSS. 因此, 窗口大小从 1 MSS 增大到 8 MSS 共需要 7 RTT.

b

由于窗口大小线性增长, 容易知道平均吞吐量为 4 MSS / RTT.