



图 8-14 如果简单盲中继是单任务的，且不支持 Connection 首部，就会挂起

这张图中发生的情况如下所述。

- 在图 8-14a 中，Web 客户端向中继发送了一条包含 Connection: Keep-Alive 首部的报文，如果可能的话要求建立一条 keep-alive 连接。客户端等待响应，以确定它要求建立 keep-alive 信道的请求是否被认可了。
- 中继收到了这条 HTTP 请求，但它并不理解 Connection 首部，因此会将报文一字不漏地沿着链路传递给服务器（参见图 8-14b）。但 Connection 首部是个逐跳首部；只适用于单条传输链路，是不应该沿着链路传送下去的。要有不好的事情发生了！
- 在图 8-14b 中，经过中继转发的 HTTP 请求抵达 Web 服务器。当 Web 服务器收到经过代理转发的 Connection: Keep-Alive 首部时，会错误地认为中继（对服务器来说，它看起来就和其他客户端一样）要求进行 keep-alive 的对话！这对 Web 服务器来说没什么问题——它同意进行 keep-alive 对话，并在图 8-14c 中回送了一个 Connection: Keep-Alive 响应首部。那么，此时，Web 服务器就认为它是在与中继进行 keep-alive 对话，会遵循 keep-alive 对话的规则。但中继对 keep-alive 会话根本就一无所知。
- 在图 8-14d 中，中继将 Web 服务器的响应报文，以及来自 Web 服务器的 Connection: Keep-Alive 首部一起发回给客户端。客户端看到这个首部，认为中继同意进行 keep-alive 对话。此时，客户端和服务器都认为它们是在进行 keep-alive 对话，但与它们进行对话的中继却根本不知道什么 keep-alive 对话。
- 中继对持久对话一无所知，所以它会将收到的所有数据都转发给客户端，等待原始服务器关闭连接。但原始服务器认为中继要求服务器将连接保持在活跃状态，所以是不会关闭连接的！这样，中继就会挂起，等待连接的关闭。
- 在图 8-14d 中，当客户端收到回送的响应报文时，它会直接转向第二条请求，在 keep-alive 连接上向中继发送另一条请求（参见图 8-14e）。简单中继通常不会期

212