|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **计算机网络与通信实验报告（二）** | | | | | | |
| 学 号 | 姓 名 | | 班 级 | | 报告日期 | |
| 181110315 | 王少博 | | 1811103 | | 2020年6月10日 | |
| 实验内容 | **利用分组嗅探器分析传输层与网络层协议** | | | | | |
| 实验目的 | 1. 分析TCP/UDP传输层协议的内部机制 2. 分析IP协议内部机制 | | | | | |
| 实验预备知识 | 1. TCP，UDP报文结构 2. IP数据包结构 3. pingplotter软件的使用 | | | | | |
| 实验过程描述 | 1. 实验环节一： 2. 下载alice.txt文件，然后上传文件但是不点击提交。 3. 打开wireshark之后，点击提交文件，得到响应的答复。 4. 筛选tcp报文 5. 实验环节二： 6. 下载软件，熟悉软件操作（这个界面和实验报告里面的差的很多，应该在option的engine里面操作，默认的时间是3所以不用改动这个，只需要调整包的大小就可以了） 7. trace的网站可以任意选择，我选择的是baidu.com 8. 调整包裹的大小，56、2000、3500之后，依次发包捕包。   总体而言和实验一的后半部分类似，都是一些读取报文进行分析的内容，但是对于一些细节会了解的更加深入，比如一些标志位的设置，例如SYNACK more fragments这些。 | | | | | |
| 实验结果 | 选取8个进行回答：   1. 实验环节一：   (3) 客户服务器之间用于初始化TCP 连接的TCP SYN 报文段的序号（sequence number）是  多少？在该报文段中，是用什么来标示该报文段是SYN 报文段的？  序号是0。  倒数第二位 Syn置1标识  (4) 服务器向客户端发送的SYNACK 报文段序号是多少？该报文段中，ACKnowledgement 字段的值是多少？Gaia.cs.umass.edu 服务器是如何决定此值的？在该报文段中，是用什么  来标示该报文段是SYNACK 报文段的？  序号为0  ACKnowledgement 为1  SYN消息中报文段序号+1得到ACK值  倒数第五位 Acknowledgement 置1标识以及倒数第二位 Syn 置1标识    (5) 包含HTTP POST 命令的TCP 报文段的序号是多少？  序号是151793    (8) 在整个跟踪过程中，接收端公示的最小的可用缓存空间是多少？限制发送端的传输以  后，接收端的缓存是否仍然不够用？      64240；没有不够用，因为发送端窗口一直是29200，小于这个大小。  (9) 在跟踪文件中是否有重传的报文段？进行判断的依据是什么？  没有。依据是有没有seq num重复。   1. 实验环节二：   (17)Identification 字段和TTL 字段的值是多少？    Time to live: 64  Identification: 0x547a (17786)  (18)所有的通过最近的路由器发送到你的电脑去的ICMP 的TTL 溢出回复是不是值都保持不  变呢？为什么？  不变。固定的路由器有固定的TTL。  再找出在pingplotter 中把包的大小改成3500 后，你的电脑所发送的第一个ICMP 请求消息。回答以下问题：  (22)从原始的数据包中产生了多少片段？片偏移分别为多少？  3段；片偏移分别为0，1480，2960 | | | | | |
| 实验当中问题  及解决方法 | 1. 实验最主要的问题和实验1后半部分一样，是存在了大量的网络问题，不过在熟悉wireshark之后就逐渐能筛选出需要的报文段了。 2. 环节一很多报文的字段需要自己点开才能看到，比如ACK SYN应答这种会有一些特定的标志位，这些积累经验之后就能找到了。 3. 环节二的分段查看时，我一开始筛选ICMP的协议数据报文来观察，无论在pingplotter中怎么修改分组大小，都没有办法看出来分段，经过一系列错误排查，发现时因为应该搜索IP协议而不是ICMP这些，最后在ICMP协议附近的IP协议字段中找到了more fragements=1的字段，并且分别对应了2个分段和3个分段的情况。 4. 关于windows size窗口大小的问题，网上很多答案回答的都不太对，这里我经过资料查询和教材上的解释，得到了如下的总结：   这里写图片描述   * 发送窗口不是发送方通告的窗口，而是接收方通告的窗口。 * 接收方通告的窗口大小=发送方窗口大小。 * 接收方通告的窗口大小也=自己的窗口大小（即接收方窗口大小） * 发送缓存>发送方窗口>已发送字节 * 接收缓存>接收窗口>接收未确认字节 * TCP通信过程中，发送缓存和接收缓存大小不变，发送窗口和接收窗口可能会改变。   TCP缓存区与窗口关系理解  **理解发送方缓存区与发送窗口的关系**  这里写图片描述  由图可知，发送缓存暂时存放：  1. 发送应用程序传给发送TCP准备发送的数据。  2. TCP已发送出但尚未收到确认的数据。  发送窗口只是发送缓存的一部分。已发送但未被确认数据大小<=发送窗口的大小。已被确认的数据会从发送缓存中删除。发送缓存和发送窗口后沿（这里的左边）是重合的，因为一被确认就会删除。删除后发送缓存和发送窗口的后沿同时向左移。发送应用程序必须控制写入缓存的速率，不能太快，否则发送缓存会没有存放数据的空间。  **理解接收方缓存区和窗口关系**  这里写图片描述  由图可知，接收缓存暂时存放：  1. 按序到达、但尚未被接收应用程序读取的数据。  2. 未按序到达的数据。  如果接收应用程序来不及读取收到的数据，接收缓存最终会被填满，使接收窗口减少到0.反之，如果接收应用程序能够及时从接收缓存中读取收到的数据，接收窗口可以增大，但最大不能超过接收缓存的大小。接收缓存和接收窗口的前沿（图的右）是重合的。因为等应用程序读取数据后，可能向前沿移动。如果接收窗口<通告时指定大小，则同时向前移。如果接收窗口不能扩大了，则不移动，接收缓存有空闲区域。记住数据只能填充接收窗口，并不是正个接收缓存 | | | | | |
| 成绩（教师打分） | 优秀 | 良好 | | 及格 | | 不及格 |