

**计算机网络**

**课程实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 实验名称 | 利用 Wireshark 进行协议分析 | | | | | |
| 姓名 | 熊峰 | | 院系 | 计算机科学与技术 | | |
| 班级 | 1903104 | | 学号 | 1190200708 | | |
| 任课教师 | 刘亚维 | | 指导教师 | 刘亚维 | | |
| 实验地点 | 格物207 | | 实验时间 | 2021.11.18 | | |
| 实验课表现 | 出勤、表现得分(10) |  | 实验报告  得分(40) |  | 实验总分 |  |
| 操作结果得分(50) |  |
| 教师评语 | | | | | | |
|  | | | | | | |

****

|  |
| --- |
| 实验目的： |
| 本次实验的主要目的：  熟悉并掌握 Wireshark 的基本操作，了解网络协议实体间进行交互以及报文交换的情况。 |
| 实验内容： |
| 概述本次实验的主要内容，包含的实验项：  1) 学习 Wireshark 的使用  2) 利用 Wireshark 分析 HTTP 协议  3) 利用 Wireshark 分析 TCP 协议  4) 利用 Wireshark 分析 IP 协议  5) 利用 Wireshark 分析 Ethernet 数据帧  选做内容：  a) 利用 Wireshark 分析 DNS 协议  b) 利用 Wireshark 分析 UDP 协议  c) 利用 Wireshark 分析 ARP 协议 |
| 实验过程： |
| 使用 Wireshark 分析 HTTP 协议，具体内容于下一部分。  使用Wireshark 分析 TCP 协议，具体内容于下一部分。  使用Wireshark 分析 IP 协议，具体内容于下一部分。  使用Wireshark 分析 Ethernet 数据帧，具体内容于下一部分。  使用Wireshark 分析 DNS 协议，具体内容于下一部分。  使用Wireshark 分析 UDP 协议，具体内容于下一部分。  使用Wireshark 分析 ARP 协议，具体内容于下一部分。 |
| 实验结果： |
| 1. Wireshark 的使用：   1.启动主机上的 web 浏览器。  2.启动 Wireshark。  3.开始分组俘获：选择WLAN网络接口，点击Start开始分组捕获。输入<http://www.hit.edu.cn>，俘获包含这些的http报文的以太网帧。  4.停止分组俘获：点击stop按键，停止分组俘获。Wireshark捕获了本机所有利用该无线网卡与其他网络实体进行交换的报文，将报文保存为Experiment4-1.pcapng。  5.筛选分组：在筛选规则中输入HTTP，分组将只显示HTTP协议报文  6.俘获的报文分析如图：     1. HTTP 分析   (1) HTTP GET/response 交互  1.启动 Web浏览器，然后启动 Wireshark 分组嗅探器。在窗口的显示过滤说明处输入“http”，分组列表子窗口中将只显示所俘获到的HTTP 报文。  2.开始 Wireshark 分组俘获。  3.在打开的Web浏览器中，输入<http://hitgs.hit.edu.cn/news>。  4.停止分组俘获。  5.俘获的报文分析如图：    思考题：  1.浏览器运行的是HTTP1.1，访问的服务器运行的是HTTP1.1。  2.浏览器向服务器指出它能接收的对象的语言版本为zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7,  zh-TW;q=0.6，接收的对象为text/html,application/xhtml+xml,application/xml;q=0.9,image/  avif,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.9\r\n。  3.本机计算机的ip地址为172.20.10.3，服务器的ip地址为202.118.254.136.  4.服务器向浏览器返回的状态代码为200.  (2) HTTP 条件 GET/response 交互  1.启动浏览器，清空浏览器的缓存。  2.启动WireShark分组俘获器，开始Wireshark分组俘获。  3.输入<http://news.hit.edu.cn>，并重新输入相同的URL并且刷新。  4.停止Wireshark分组俘获，并使用http筛选。  俘获的报文分析如图：    思考题：  1.经过大量尝试，尝试<http://www.hit.edu.cn> <http://cs.hit.edu.cn> <http://news.hit.edu.cn> 等网页都没有找到存在IF-MODIFIED-SINCE的请求报文。    2.服务器明确返回了文件的内容。如图可以看到，服务器传回了文件内容。  3.经过大量尝试后，没有找到IF-MODIFIED-SINCE字段。    4.仍然传回200 OK，并且明确传回了文件内容。   1. TCP 分析   A.俘获大量的由本地主机到远程服务器的 TCP 分组  (1)启动浏览器，打开<http://gaia.cs.umass.edu/Wireshark-labs/alice.txt>网页，得到ALICE'S ADVENTURES IN WONDERLAND文本，将该文件保存到你的主机上。  (2)打开<http://gaia.cs.umass.edu/Wireshark-labs/TCP-Wireshark-file1.html>，并选择保存的txt。  (3)启动Wireshark，开始分组俘获。  (4)在浏览器中上传文件。  (5)停止俘获。报文为Experiment3-1.pcapng.    B.浏览追踪信息  在筛选规则中输入tcp and ip.addr == 128.119.245.12，浏览本地主机和服务器之间传输的tcp和http报文，以及主机向服务器发送的HTTP POST报文和一系列的”http continuation”报文。  思考题：  1.客户端主机的ip地址为172.20.10.3，TCP端口号是1379.  2.gaia.cs.umass.edu服务器的IP地址是128.119.245.12，接收的端口号为80.  C.TCP基础    1.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0，在该报文段中，用1来标识Syn段表示是SYN报文段。    2.服务器向客户端发送的SYN为1，ACK为1，报文段的序号是sequence number是0，Acknowledgement字段的值为1，Acknowledgement字段通过对SYN报文段的sequence number+1获得。    3.以上三张图分别为三次握手的过程。分别是客户端向服务器端发送SYN请求报文；服务器向客户端回复SYN ACK报文；客户端向服务器回复ACK报文。      4.如上图，通过计算seq和length，不难得出，第六个报文段的序号为37，发送时间为第一帧发送后的2.654315s，接收的时间为18:25:17.79778000。    5.如图，前六次的报文的长度依次为648、12600、1400、5600、19600、2800.    6.整个跟踪过程中，接收端公示最小可用缓存空间是 29200。我们可以看到当限制了发送方的传输后，接收端缓存足够。    7.如上图，对TCP流报文追踪，发现无重传片段。        8.对所有TCP流追踪，整个对话共153kB，总时间为2.643255s，因此throughput=152935/ 2.643255=57,858.58723430013(字节每秒)。   1. IP 分析     以[www.zhihu.com](http://www.zhihu.com)为例，使用PingPlotter，分别以pack size = 2000，和pack size = 3500运行。并启用Wireshark进行报文获取。    (1)  思考题：    1.主机的IP地址为172.20.10.3.    2.在IP数据报头中，上层协议字段的值是1，标识ICMP，IP数据报头中上层协议字段的值为1表示ICMP协议。    3.IP头有20B，总长为520B，IP数据包的净载大小为500B.确定的方式为：IP数据包总长度-IP数据包头部长度。    4.该数据包未分片，由于标志位全为0，表示允许分片但未分片。  (2)    思考题：  1.主机发出的 ICMP 报文中中 IP 数据报一些字段总在发生改变：标识 ID、生存时间、首部校验和、数据域。  2.除了以上四个数据外，其余的数据保持常量，原因是：标识 ID 唯一，所以每个数据报有所区别，随之首部校验和也不断改变；TTL 在不断变大（因为是 ICMP 的 ping 探测），而且数据域中封装有 ICMP 的报文，因为 ICMP 的头部信息也在变化，所以 IP 数据报的数据域也随之变化。  3. IP 数据报的 Identification 字段值为每一个报文一个唯一的 16b 的数值，且在线性递增，不断执行加一操作。  (3)    思考题：  1.Identification字段的值为0x78e4，TTL字段的值是64.  2.Identification字段变化，区分不同的ICMP time-to-live exceeded消息，TTL保持不变，均为一次转发。  (4)    思考题：  1.该消息被分解成不止一个数据报。    2.标志位MF表明后面还有分片。如上图，该分片的数据域长度为1480B，IP总长度为1500B.    3.如上图，在改为3500字节后，原始数据包被分成了3片。  4.标志位和checksum字段发生了变化。   1. 抓取Ethernet数据包   1.访问[www.hit.edu.cn](http://www.hit.edu.cn)进行抓包分析。  2.主机收线发送给第一条请求HTTP报文，以太网帧结构封装了上层的IP数据，IP封装了上层的TCP数据报，TCP数据报封装了上层的HTTP数据包。   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 以太网帧首部 | IP首部 | TCP首部 | HTTP请求报文 | CRC |   3.以太网帧的结构如下：  1)目的MAC、源MAC地址(各6B):若网卡的MAC地址与收到的帧的目的MAC地址匹配，或者帧的目的 MAC 地址为广播地址(FF-FF-FF-FF-FF-FF)，则网卡接收该帧，并将其封装的网络层分组交给相应的网络层协议；否则，网卡丢弃(不接收)该帧。  2)类型 Type2B：指示帧中封装的是哪种高层协议的分组 (如，IP 数据报、Novell IPX 数据报、AppleTalk 数据报等)。  3)数据(Data)(46-1500B): 指上层协议载荷。  4)CRC(4B): 循环冗余校验码，丢弃差错帧。   1. 抓取 ARP 数据包   (1)本机的ARP缓存如下：      (2)以管理员方式运行命令行提示符，输入 arp -d 删除本机 ARP 缓存，使用ping 172.20.10.2命令，获得ARP数据包。  ARP数据包的组成如下：硬件类型 2B、协议类型 2B、硬件地址长度 1B、协议地址长度 1B、OP2B、源 MAC 地址 6B、源 IP 地址 4B、目的 MAC 地址 6B、目的 IP 地址 4B.   |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 硬件类型 | 协议类型 | 硬件地址长度 | 协议地址长度 | OP | 发送端MAC地址 | 发送端IP地址 | 目标MAC地址 | 目标IP地址 |   0 2 4 5 6 8 14 18 24 28  (3)通过OP，当OP值为1时是请求包，当OP值为2时是应答包。    (4) 查询ARP没有IP 对应的MAC地址，因此需要广播查询，即设置目的 MAC 地址为 ff: ff:ff:ff:ff:ff， ARP 由于在接收到的查询ARP 中找到了源 MAC 地址，因此响应有一个明确的目的地址。   1. 抓取 UDP 数据包     (1)消息是基于UDP协议的。    (2)主机IP是172.20.10.3，目的主机IP是157.148.55.161.    (3)主机发送QQ消息的端口号是4006，QQ服务器的端口号是8000.    (4)源端口号 2B，目的端口号 2B，UDP 段长度 2B，校验和 2B。   |  |  |  |  | | --- | --- | --- | --- | | 源端口号 | 目的端口号 | UDP长度 | UDP校验和 |   0 2 4 6 8  (5)因为UDP是不可靠的无连接的传输服务，服务器返回ICQ作为确认，客户端通过返回的ICQ确认服务器已经收到了数据报，所以需要服务器返回ICQ报文。证明UDP 是无连接的。因为TCP报文需要三次握手建立连接，而且需要TCP报文段首部中的标志位，但是UDP首部无标志位，UDP也无序列号。通过抓包分析UDP的数据结构可以判断UDP是无连接的。   1. 利用 WireShark 进行 DNS 协议分析   以www.baidu.com为例，使用Wireshark抓包分析。    (1)主机IP为172.20.10.3。  (2)DNS消息格式如下：    (3)DNS使用Transaction ID来标识查询和响应，其大小为2B，一次请求和对应的响应ID是一致的。 |
| 问题讨论： |
| 1.ARP数据包的获取，后续实验中已解决。  2.实验未出现IF-MODIFIED-SINCE字段。 |
| 心得体会： |
| 对五层模型的体会更加深刻，自顶向下系统的复习了互联网五层模型。对各层的应用都有所了解，对应用层的HTTP协议，传输层的TCP、IP，网络层的IP、路由协议，数据链路层的以太网等了解的更加深刻。 |