**深 圳 大 学 实 验 报 告**

**课程名称：­ 计算机网络**

**实验项目名称： 实验3 数据包抓取与分析**

**学院： 数学与统计学院**

**专业： 信息与计算科学（数学与计算机实验班）**

**指导教师： 王璐**

**报告人： 王曦 学号：2021192010 班级： 数计班**

**实验时间： 2023年03月09日**

**实验报告提交时间： 2023年03月09日**

**教务处制**

|  |  |
| --- | --- |
| **实验目的**  学习安装、使用协议分析软件，掌握基本的数据报抓取、过滤和分析方法，能分析 HTTP、TCP、ICMP 等协议。 | |
| **实验环境**   1. 使用具有Internet 连接的Windows 操作系统； 2. 抓包软件Wireshark。 | |
| **实验内容：**   1. 安装学习Wireshark 软件； 2. 抓包与分析HTTP 协议； 3. 分析TCP 协议； 4. 分析TCP 三次握手； 5. 分析ICMP 协议。 |
| **实验步骤：**  （用文字描述实验过程，并用截图辅助说明）   1. **下载并安装Wireshark**   Wireshark是世界上最广泛使用的网络协议分析器。  从官网https://www.wireshark.org/下载并安装Wireshark.  1.1 运行Wireshark, 初始界面如图1所示. 从接口列表中选择要捕获的接口, 双击开始捕获. 下面以捕获以太网为例.    图1: Wireshark的初始界面  1.2 如图2, 进入Wireshark主界面, 开始捕获分组.    图2: Wireshark的主界面  其中分组详情栏的解释如图3.    图3: Wireshark的分组详情栏的解释  1.3 使用过滤器  过滤器包括协议过滤、IP地址过滤、模式过滤、端口过滤.   1. 协议过滤   以过滤HTTP协议为例. 如图4, 在过滤器中输入”http”并回车, 观察到分组列表栏中的条目的Protocol都为HTTP.  注意上述操作只会过滤使用HTTP协议的条目, 而不会过滤HTTPS协议的条目.    图4: HTTP协议过滤   1. IP地址过滤   ①如图5, 在Powershell中ping百度的网址, 得到其IP地址182.61.200.7.    图5: 获得百度的IP地址  ②如图6, 在Powershell中用ipconfig得到本地的IP地址172.31.68.104.    图6: 获得本地的IP地址  ③打开Wireshark捕获分组, 用浏览器访问”www.baidu.com”后停止捕获. 如图7, 观察到分组列表栏中的条目的源IP地址都为本地的IP地址, 目的地的IP地址都为百度的IP地址.    图7: IP地址过滤   1. 模式过滤   过滤分组列表栏中的http情求方法为GET的分组. 如图8, 观察到条目的Info中以GET开头.    图8: 模式过滤   1. 端口过滤   过滤分组列表兰中tcp端口号为80的分组. 如图9, 观察到条目的Info中的端口号都为80.    图9: 端口过滤   1. **抓包与分析HTTP协议**    1. 打开Wireshark捕获分组, 在过滤器中输入”http”以过滤http协议的分组.    2. 打开浏览器, 输入一个以前未访问过的网址, 如”https://codeforces.com/”, 访问的网址需以前未访问过是为了防止浏览器直接调用缓存, 而不使用http协议获取数据.    3. 如图10, 观察到分组列表栏中出现HTTP协议的分组.     图10: HTTP协议过滤  2.4 逐个查看分组详情栏, 检查哪些条目是浏览网页产生的分组. 如图11, 分析得前两个分组是浏览网页产生的, 可从分组列表栏中得知此次浏览网页的源IP地址和目的地IP地址, 从分组详情栏得知此次浏览网页的用户名和主机名.    图11: 浏览网页产生的分组   1. **分析TCP协议**   3.1 分析2.4中的分组的TCP协议信息. 如图12, 在分组详情栏中得知本次TCP连接的源端口号、目的端口号、序列号、确认号、报头长度、标志位、窗口大小、校验和、数据.    图12: TCP协议的详情  3.2 追踪上述分组的TCP流. 如图13, 右键该条目, 追踪流-TCP流.    图13: 追踪TCP流  3.3 如图14, 找到Info中带有[SYN]、[SYN, ACK]、[ACK]的前三个分组, 发现它们在HTTP GET请求之前, 它们是TCP建立的分组.  原理: ①TCP建立在HTTP GET请求前; ②TCP建立连接时会设置标志位SYN.    图14: TCP建立的分组   1. **分析TCP三次握手**     图15: TCP三次握手示意图  4.1 TCP第一次握手(SYN), 在分组详情栏中得知序号3951008536, 且Flags中的Syn位置1.    图16: TCP第一次握手(SYN)  4.2 TCP第二次握手(SYNACK), 在分组详情栏中得知序号、确认、ACK位置1、SYN位置1. 观察到确认号 = SYN序号 + 1.    图17: TCP第二次握手(SYNACK)  4.3 TCP第三次握手(ACK), 在分组详情栏中得知确认号、ACK位置1、SYN位置0. 观察到确认号 = SYNACK序号 + 1.    图18: TCP第三次握手(ACK)   1. **分析ICMP协议**    1. 打开Wireshark捕获分组, 在过滤器中输入”icmp”以过滤icmp协议的分组. icmp协议是ping命令使用的协议.    2. 如图19, 在Powershell中ping百度的网址.   图19: ping百度的网址   * 1. 如图20, 在分组列表栏中观察到4次ping的请求和回复.     图20: 过滤ICMP协议 |
| **实验结果：**  （给出个人对结果的分析、结论）  数据包的抓取和协议的分析结果见实验步骤部分.  下面对实验内容的TCP协议部分做一些补充.   1. **TCP协议示意图**   TCP协议示意图如图21所示, 其中第二层和第三层即图12中所见的序列号和确认号.    图21: TCP协议示意图   1. **TCP的三次握手**   TCP的三次握手示意图如图22所示.    图21: TCP三次握手示意图  2.1 TCP第一次握手(SYN), 如图22, 观察到客户端将同步号SYN位置1, 并选择初始序列号x, 将该数据报发送给服务器, 客户端进入SYN\_SENT状态, 等待服务器确认.    图22: TCP第一次握手(SYN)  2.2 TCP第二次握手(SYNACK), 如图23, 观察到服务器接收到数据报后, 由SYN = 1得知客户端请求建立连接, 服务器将ACK位和SNY位都置1, 确认号置为(x + 1), 并选择初始序号y, 将该数据报发送给客户端以确认连接请求, 服务器进入SYN\_RCVD状态.    图23: TCP第二次握手(SYNACK)  2.3 TCP第三次握手(ACK), 如图24, 观察到客户端接收到确认信息后, 检查ACK位是否为1和确认号是否为(x + 1), 若正确则将ACK位置1, 将确认号设为(y + 1), 将该数据报发送到服务器. 服务器检查ACK = 1和确认好 = (y + 1)是否都成立, 若是则TCP连接建立成功, 客户端和服务器都进入ESTABLISHED状态, 三次握手完成, TCP连接建立, 可进行稳定可靠的数据传输.    图24: TCP第三次握手(ACK)   1. **利用TCP三次握手的DDoS攻击**   DDos攻击的示意图如图25所示.  IMG_256  图25: DDoS攻击示意图  攻击者通过操控大量僵尸机器(Zombies), 大量向DNS发送目标IP地址的请求, 在DNS服务器与目标主机建立TCP连接时, 攻击者只进行第一、二次握手, 服务器持续等待第三次握手的确认, 造成资源和数据库池的占用, 无暇应答正常访问的用户. |
| **实验小结：**  （实验中出现问题的解决方法，实验心得体会等）   1. 本次实验安装了Wireshark软件, 学习了用Wire实现抓取和分析数据包. 具体地, 本次实验分析了HTTP协议、TCP协议、TCP三次握手和ICMP协议. 2. 通过参考助教提供的实验报告样例, 我学习到了用表格的方式理解TCP的三次握手, 如下:   (1)第一次握手(SYN): Client发送SYN数据, 在收到Server的SYN/ACK数据前无法确认己方发送是否正常, 此时Client的状态如下:   |  |  |  |  |  | | --- | --- | --- | --- | --- | | Client (1) | 己方发送正常 | 己方接收正常 | 对方发送正常 | 对方接收正常 | | 第一次握手 |  |  |  |  | | 第二次握手 |  |  |  |  | | 第三次握手 |  |  |  |  |   Server确认后可确定己方接收正常、对方发送正常, 此时Server状态如下:   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Server (2) | 己方发送正常 | 己方接收正常 | 对方发送正常 | 对方接收正常 |  | | 第一次握手 |  | √ | √ |  |  | | 第二次握手 |  |  |  |  |  | | 第三次握手 |  |  |  |  |  |   (2)第二次握手(SYN/ACK): Client确认己方发送正常、己方接收正常、对方发送正常、对方接收正常. 此时Client确认状态完成, 状态如下:   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Client (2) | 己方发送正常 | 己方接收正常 | 对方发送正常 | 对方接收正常 |  | | 第一次握手 |  |  |  |  |  | | 第二次握手 | √ | √ | √ | √ |  | | 第三次握手 |  |  |  |  |  |   Server只能确认己方接收正常、对方发送正常, 状态如下:   |  |  |  |  |  | | --- | --- | --- | --- | --- | | Server (2) | 己方发送正常 | 己方接收正常 | 对方发送正常 | 对方接收正常 | | 第一次握手 |  |  |  |  | | 第二次握手 |  | √ | √ |  | | 第三次握手 |  |  |  |  |   (3)第三次握手(ACK): Client确认状态在第二次握手时已完成. Server在第三次握手可确认己方发送正常和对方接收正常, Server确认状态完成, 状态如下:   |  |  |  |  |  | | --- | --- | --- | --- | --- | | Server (3) | 己方发送正常 | 己方接收正常 | 对方发送正常 | 对方接收正常 | | 第一次握手 |  |  |  |  | | 第二次握手 |  |  |  |  | | 第三次握手 | √ | √ | √ | √ |   至此, 需要且仅需要3次握手即可确认连接是否正常, 进而可保证稳定可靠的数据传输. |
| **指导教师批阅意见：**  **成绩评定：**  **指导教师签字：**  年 月 日 |
| 备注： |