网络与信息安全课内实验一

对 DDos 攻击的理解

1. 实验目的

DDoS 攻击指分布式拒绝服务攻击，即处于不同位置的多个攻击者同时向一个或数个目标发起攻击，或者一个攻击者控制了位于不同位置的多台机器并对受害者同时实施攻击。DDoS 有很多种类型 SYN Flood 攻击、TCP 全连接攻击、TCP 混乱数据包工具、UDP Flood 攻击、DNS Flood 攻击、CC 攻击等，其中最常见的也是最经典的就是 SYN Flood 攻击。

本次实验的目的就是通过实践，加深对 SYN 泛洪攻击的理解。

2. 实验原理

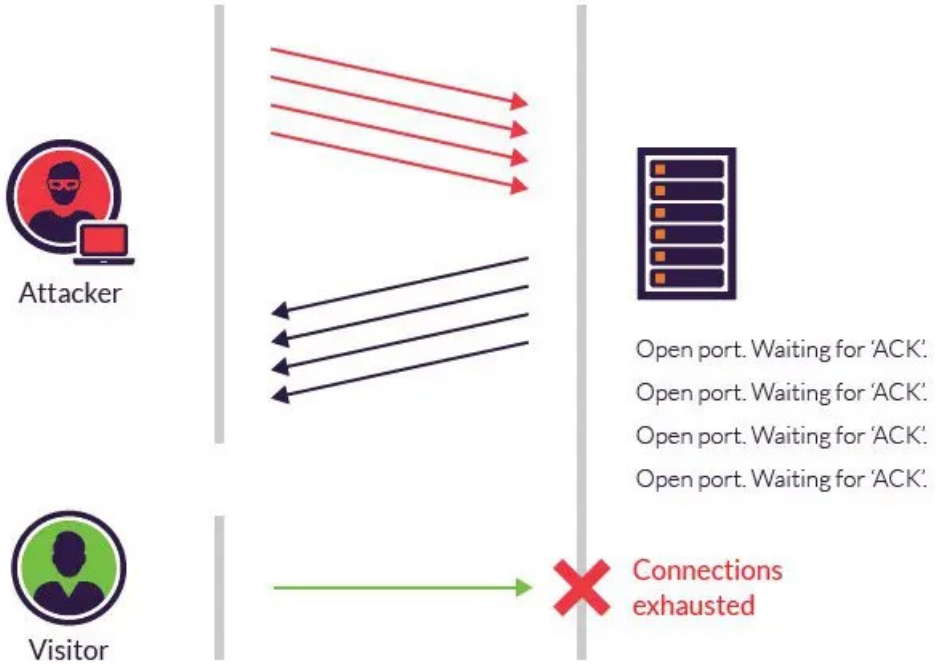
SYN 泛洪攻击利用了这 TCP 协议实现的一个缺陷：

在 TCP 三次握手的过程中，服务器为了响应一个收到的 SYN，要分配并初始化连接变量和缓存，用两端的 IP 地址与端口号来标识这个 TCP 包属于哪个 session。使用一张表来存储所有的 session，表中的每条称作 Transmission Control Block（TCB），TCB 结构的定义包括连接使用的源端口、目的端口、目的 IP、序号、应答序号、对方窗口大小、己方窗口大小、TCP 状态、TCP 输入/输出队列、应用层输出队列、TCP 的重传有关变量等，当然 TCB 要占用一定的内存。

然后服务器发送一个 SYN-ACK 进行响应，并等待来自客户的 ACK 报文段。如果某个客户不发送 ACK 来完成该三次握手的的第三步，服务器端将重发 SYN-ACK，在 Linux 下，默认重试次数为 5 次，重试的间隔时间从 1s 开始每次都翻倍，5 次的重试时间间隔为 1s, 2s, 4s, 8s, 16s，总共 31s，第 5 次发出后还要等 32s 才知道第 5 次也超时了，所以，总共需要 1s + 2s + 4s+ 8s+ 16s + 32s = 63s，服务器才会终止该半开连接并回收资源，这几个参数也可以调整。

在此期间，服务器无法通过发送 RST 数据包来关闭连接，连接保持打开状态。 在连接超时之前，另一个 SYN 数据包将到达。随着大量的 SYN 报文段涌入，服务器不断为这些半开连接分配资源，TCP 套接口缓存队列被迅速填满，而拒绝新的连接请求从而阻止其他合法用户进行访问。

我们此次实验采取的是 IP 地址欺骗，而不是真正的分布式攻击，随机生成 RandIP，指定目标 IP，指定目标端口以及 TCP 标志位，scapy 库帮助我们重新生成原始 IP 报文。要使攻击成功，位于伪装 IP 地址上的主机必须不能响应任何发送给它们的 SYN-ACK 包。



3. 实验步骤

准备两台 ubuntu 16.04 虚拟机，处于同一局域网，一台当 server，一台当 attacker，两台机器的网络都改为桥接模式

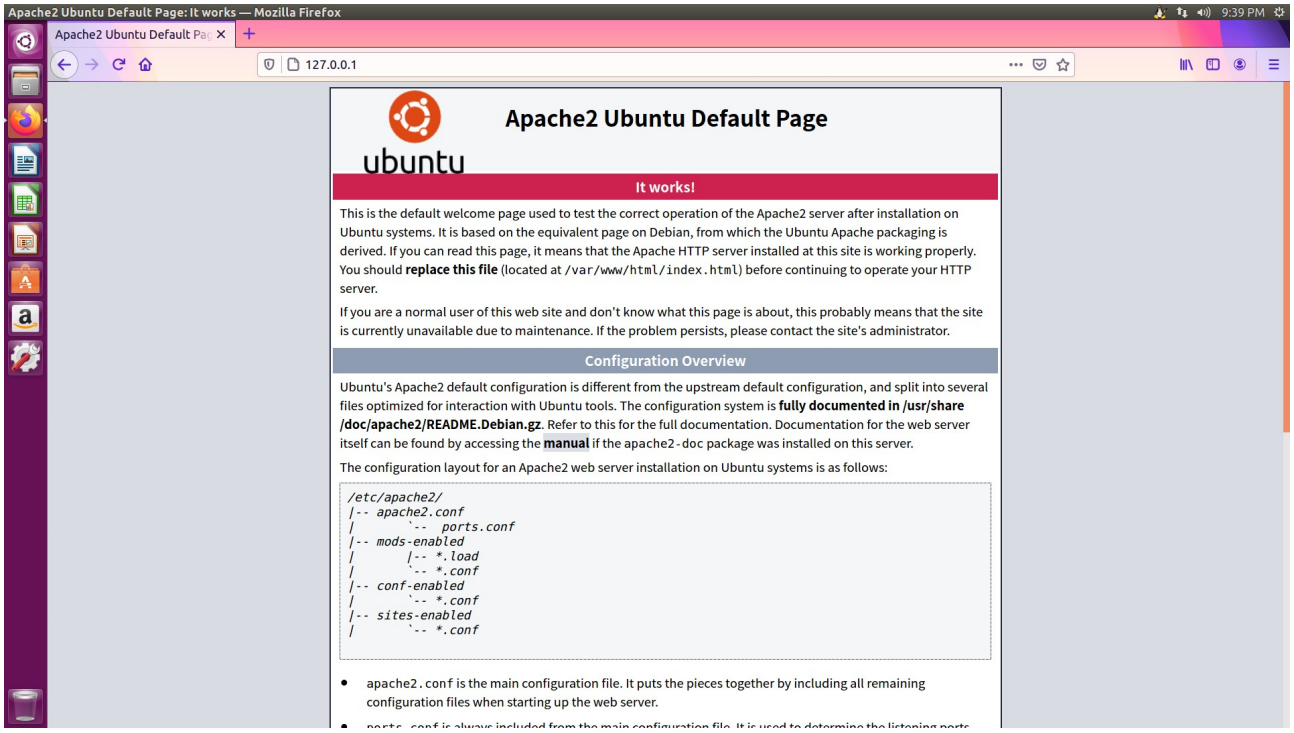
在 server 上安装 Web 服务器软件 Apache2，搭建一个简单的网站，作为被攻击的目标。安装 wireshark 进行抓包分析。

server 的 IP 地址为 192.168.1.102

```
justin@ubuntu-server:~
justin@ubuntu-server:~$ ifconfig
ens33     Link encap:以太网  硬件地址 00:0c:29:4c:66:5d
          inet 地址:192.168.1.102  广播:192.168.1.255  掩码:255.255.255.0
          inet6 地址: fe80::4fe5:8aba:df2d:870f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  跃点数:1
          接收数据包:100904 错误:0 丢弃:0 过载:0 帧数:0
          发送数据包:59928 错误:0 丢弃:0 过载:0 载波:0
          碰撞:0 发送队列长度:1000
          接收字节:6736772 (6.7 MB)  发送字节:3813351 (3.8 MB)

lo        Link encap:本地环回
          inet 地址:127.0.0.1  掩码:255.0.0.0
          inet6 地址: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  跃点数:1
          接收数据包:566 错误:0 丢弃:0 过载:0 帧数:0
          发送数据包:566 错误:0 丢弃:0 过载:0 载波:0
          碰撞:0 发送队列长度:1000
          接收字节:57153 (57.1 KB)  发送字节:57153 (57.1 KB)
```

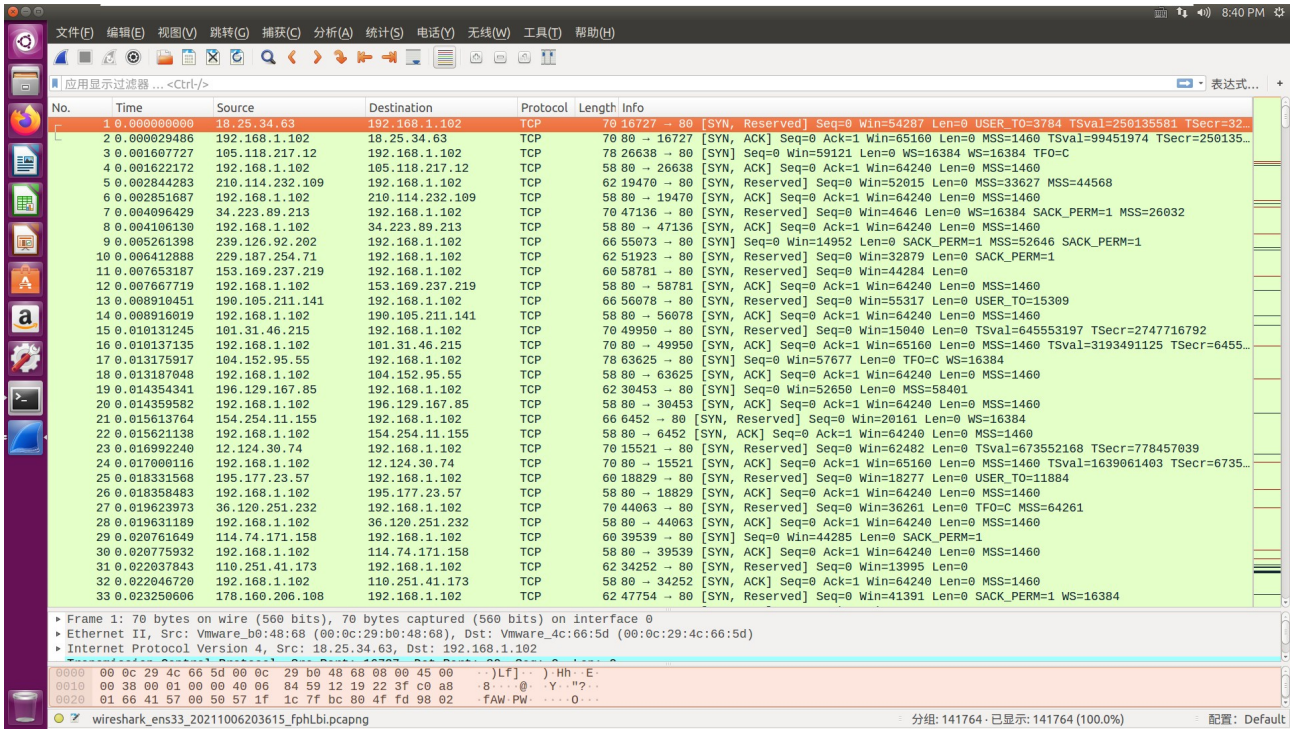
这是默认的网页，局域网内其他设备可以通过 192.168.1.102 访问



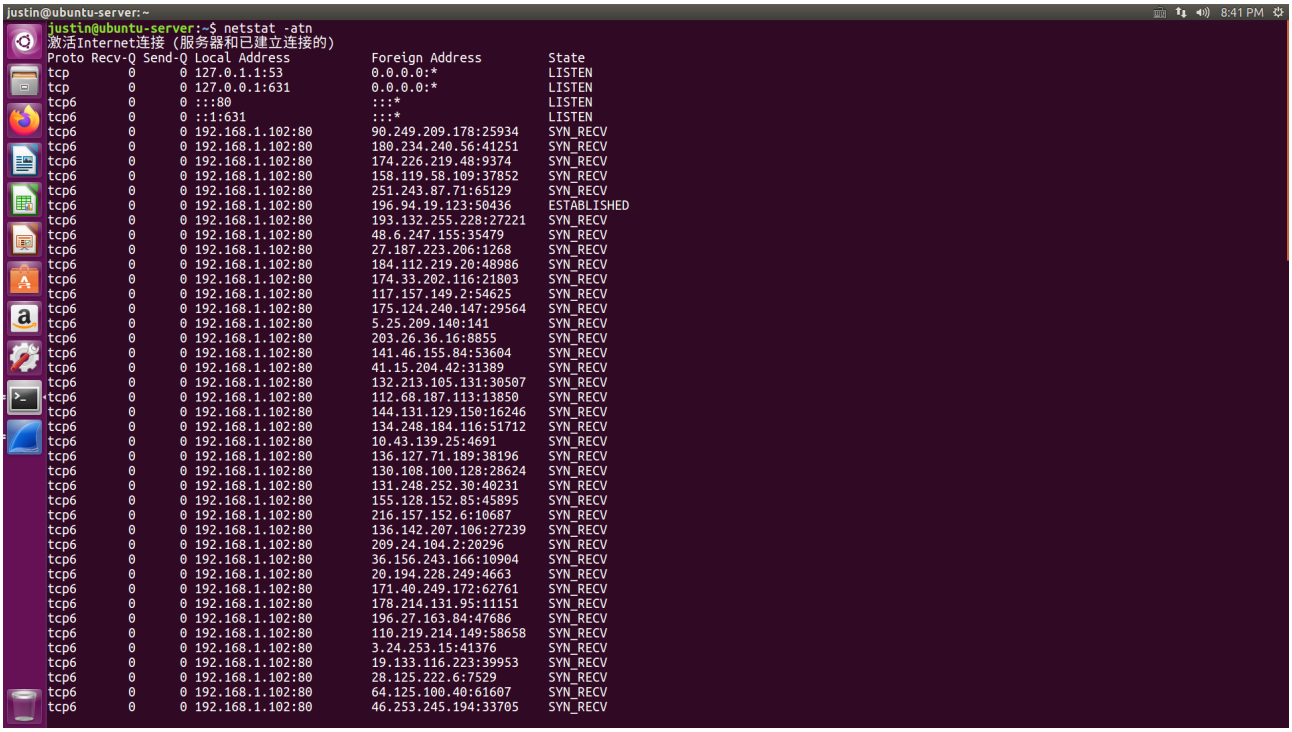
在 server 上运行 apache 服务器

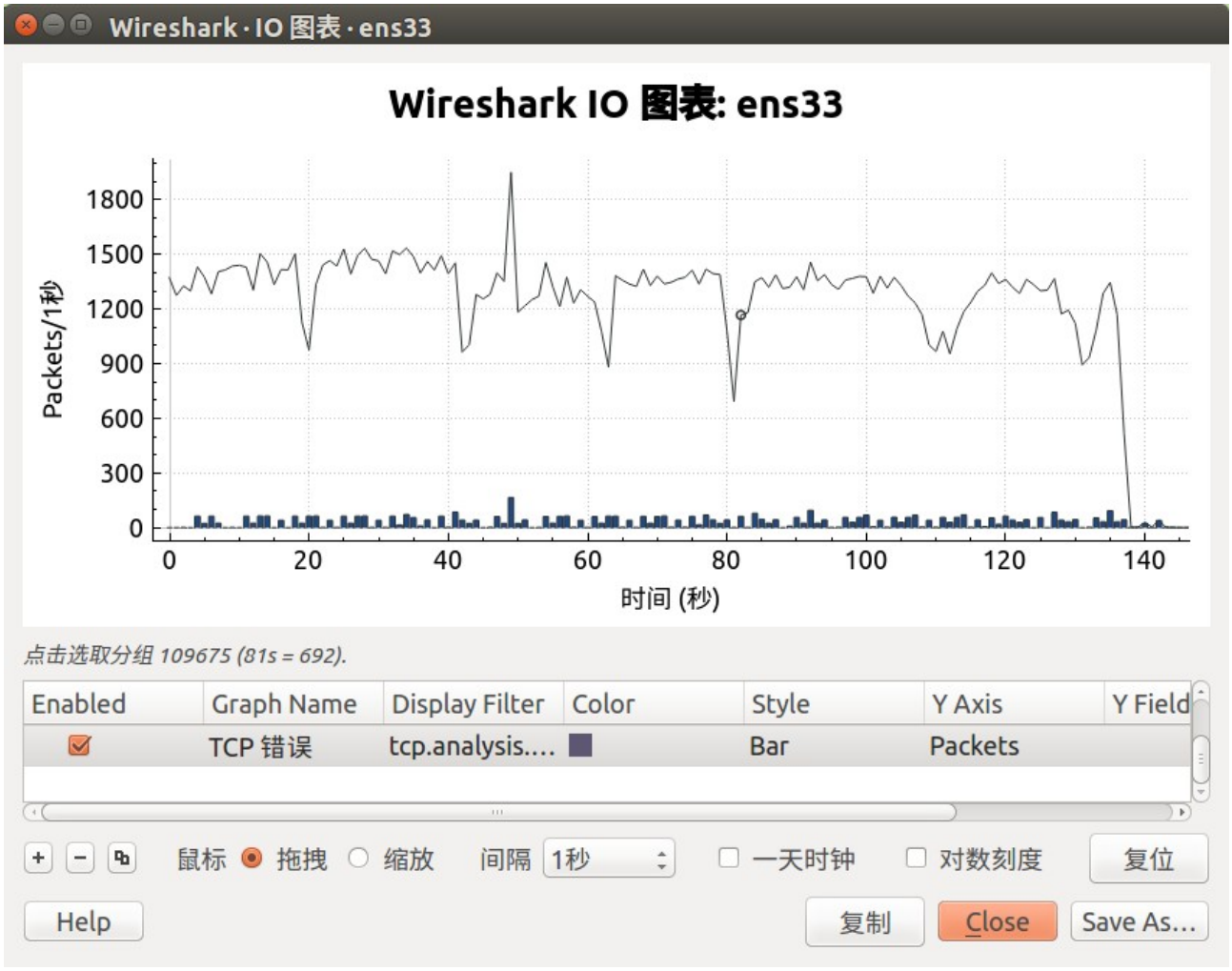
在 attacker 上安装 python 库 scapy，并编写程序攻击 server。

开始攻击！！！



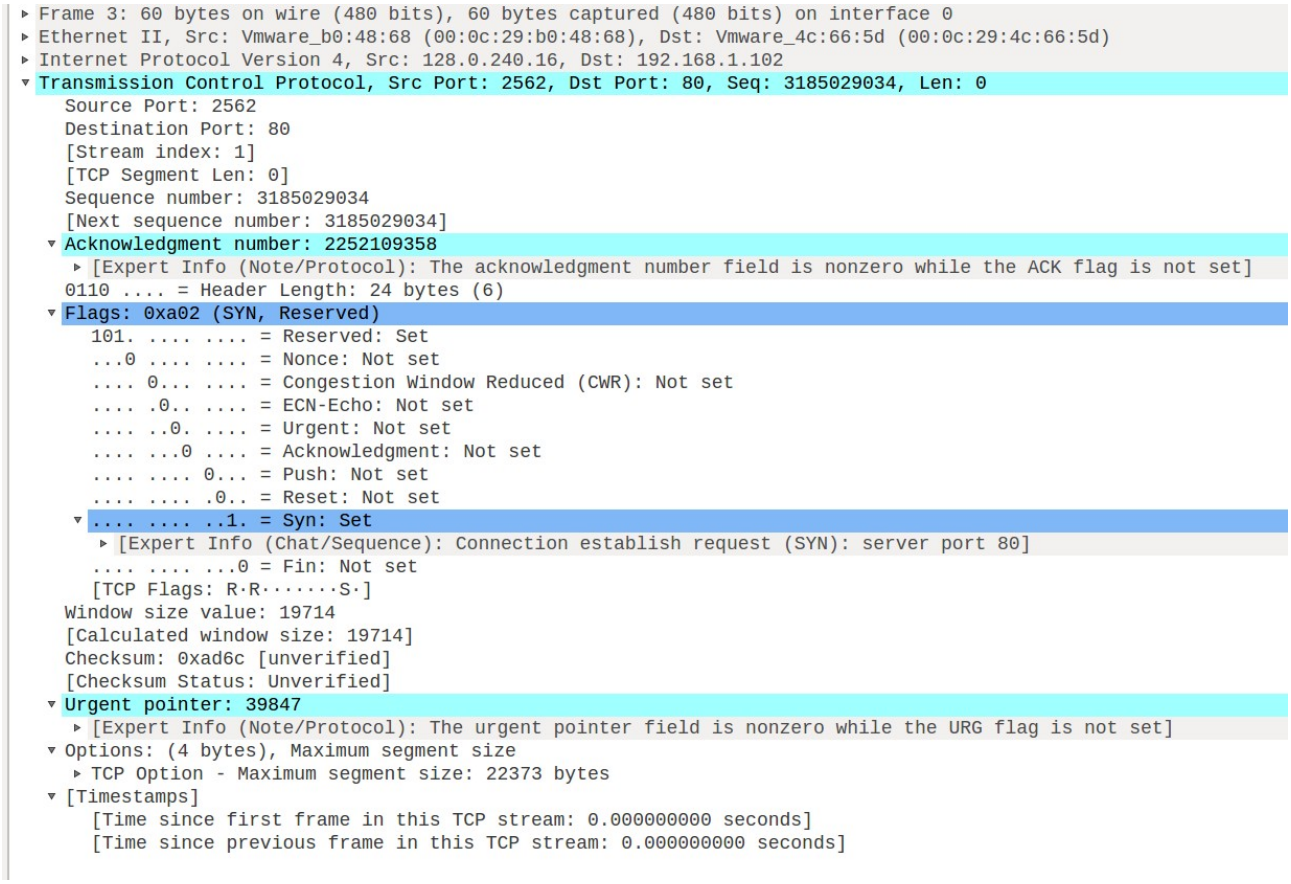
我们可以看到，客户端和服务器端 TCP 连接只握手了两次，都处于半开连接状态





纵使有如此猛烈的 SYN 泛洪攻击，我们的服务器依然可以正常提供服务。

TCP 参数解释，以下面这两次为例



客户端的 TCP 首先向服务器端的 TCP 发送一个特殊的 TCP 报文段。

源端口：2562，目标端口：80，客户端分配一个端口，服务器端 80 端口提供 HTTP 服务。用来标识同一台计算机的不同的应用进程，使客户端和服务器端能进行进程间的逻辑通信

序列号：3185029034，在 TCP 连接中，所传送的字节流的每一个字节都会按顺序编号。当 SYN 标记不为 1 时，这是当前数据分段第一个字节的序列号；当 SYN 标记为 1 时，这个字段的值就是初始序列值（ISN），由客户端随机生成

确认号：确认号只有当 ACK 标志为 1 时才有效，因此此时的确认号是无效的。表示接收方期望收到发送方下一个报文段的第一个字节数据的编号，其值是接收方即将接收到的下一个序列号

首部长度/数据偏移：24bytes，表明 TCP 报文段头部长度为 24bytes，同时告诉接收端的应用程序，数据从何处开始

保留字段：占 4 位，为 TCP 将来的发展预留空间，目前必须全部为 0

标志位字段： 当 SYN=1，ACK=0 时，表示这是一个请求建立连接的报文段

- CWR（Congestion Window Reduce）：拥塞窗口减少标志，用来表明它接收到了设置 ECE 标志的 TCP 包。并且，发送方收到消息之后，通过减小发送窗口的大小来降低发送速率。
- ECE（ECN Echo）：用来在 TCP 三次握手时表明一个 TCP 端是具备 ECN 功能的。在数据传输过程中，它也用来表明接收到的 TCP 包的 IP 头部的 ECN 被设置为 11，即网络线路拥堵。

- URG（Urgent）：表示本报文段中发送的数据是否包含紧急数据。URG=1 时表示有紧急数据。当 URG=1 时，后面的紧急指针字段才有效。
- ACK：表示前面的确认号字段是否有效。ACK=1 时表示有效。只有当 ACK=1 时，前面的确认号字段才有效。TCP 规定，连接建立后，ACK 必须为 1。
- PSH（Push）：告诉对方收到该报文段后是否立即把数据推送给上层。如果值为 1，表示应当立即把数据提交给上层，而不是缓存起来。
- RST：表示是否重置连接。如果 RST=1，说明 TCP 连接出现了严重错误（如主机崩溃），必须释放连接，然后再重新建立连接。
- SYN：在建立连接时使用，用来同步序号。当 SYN=1，ACK=0 时，表示这是一个请求建立连接的报文段；当 SYN=1，ACK=1 时，表示对方同意建立连接。SYN=1 时，说明这是一个请求建立连接或同意建立连接的报文。只有在前两次握手中 SYN 才为 1。
- FIN：标记数据是否发送完毕。如果 FIN=1，表示数据已经发送完成，可以释放连接。

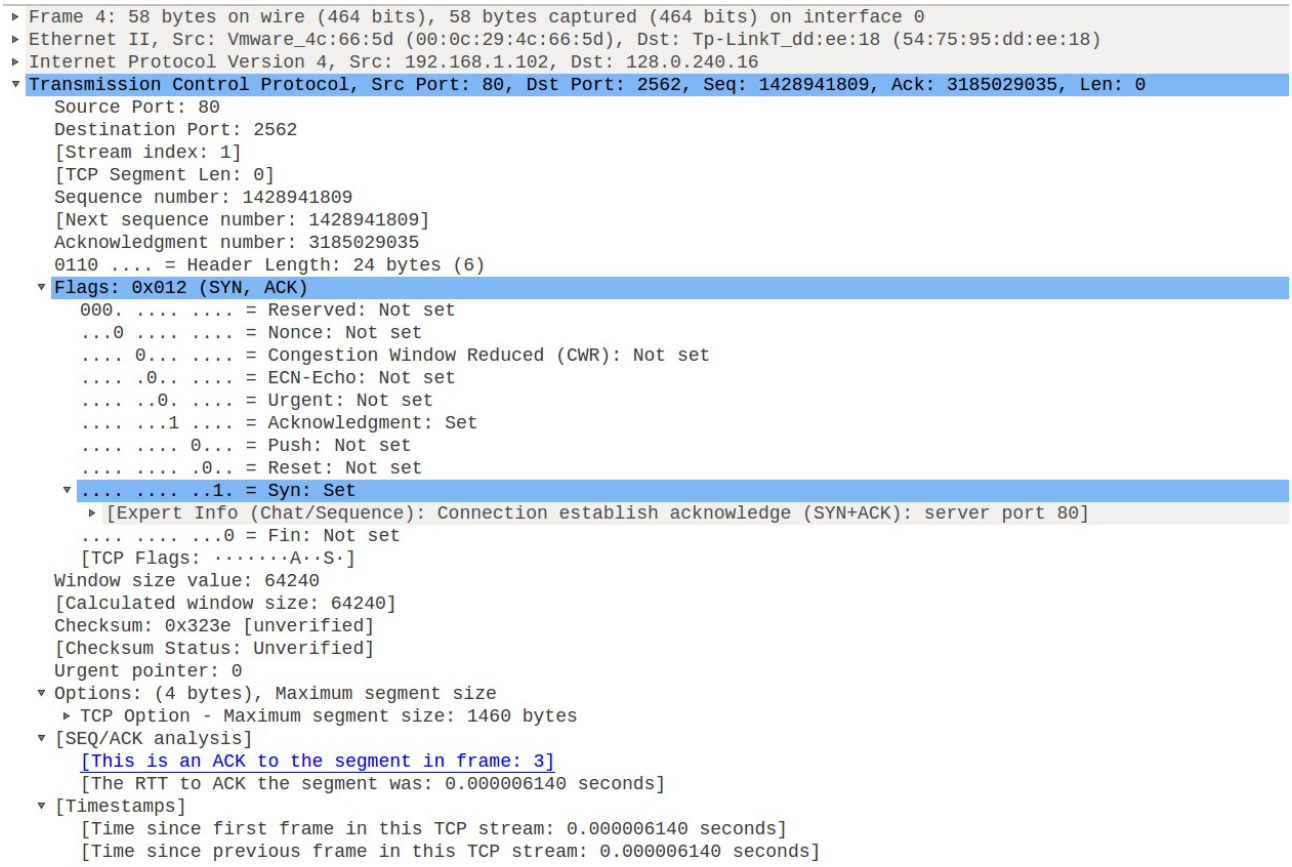
窗口大小字段：19714bytes，告知发送端当前接收端的接收窗口还有多少剩余空间，以此控制发送端发送数据的速率，起到流浪控制的作用

校验和字段：0xad6c，对整个 TCP 报文段的校验，包括 TCP 头部和 TCP 数据，用 checksum 算法得到。

发送端基于数据内容校验生成一个数值，接收端根据接收的数据校验生成一个值，如果两个值不同，则丢掉这个数据包重发

紧急指针：无效，仅当前面的 URG 控制位为 1 时才有意义。是一个正的偏移量，和顺序号字段中的值相加表示紧急数据最后一个字节的序号。当所有紧急数据处理完后，TCP 就会告诉应用程序恢复到正常操作。即使当前窗口大小为 0，也是可以发送紧急数据的，因为紧急数据无须缓存

选项字段：指定了应用层数据最长为 22373bytes，又称为 MSS（Maximum Segment Size）。每个连接方通常都在通信的第一个报文（为建立连接而设置 SYN 标志位为 1 的那个段）中指明这个选项，它表示本端所能接收的最大的应用层数据的长度。选项字段不一定是 32bits 的整数倍，因此需要加填充位填充



源端口：80，目标端口：2562

序列号：1428941809，这是服务器端随机生成的初始序号

确认号：3185029035，已经收到客户端序号 3185029034

首部长度/数据偏移：24bytes

保留字段：占 4 位，为 TCP 将来的发展预留空间，目前必须全部为 0

标志位字段： 当 SYN=1，ACK=1 时，表示这是一个同意建立连接的报文段

窗口大小字段：64240bytes

校验和字段：0x323e

紧急指针：无效， 仅当前面的 URG 控制位为 1 时才有意义

选项字段：指定了应用层数据最长为 1460bytes，也就是 TCP 数据包每次能够传输的最大数据分段，称为 MSS（Maximum Segment Size），看英文名比较容易误解

一旦包含 TCP SYN 的报文端的 IP 数据报到达服务器主机，服务器会从该数据报中抽取出 TCP SYN 字段，为该 TCP 连接分配 TCP 缓存和变量，此过程中我们的服务器主机就分配了 64240bytes 的窗口，并向该客户 TCP 发送允许连接的报文段

4. 实验小结

这次实验让我熟悉了 linux 操作系统，有很多实用的命令，
cd、ls、sudo、su、mkdir、cp、dig、ifconfig、ping、netstat、traceroute 等等，还学会了用 vim 写代码然后命令行编译运行

同时也熟悉 wireshark 的使用，通过逐字节分析报文深入网络原理

详细了解了 TCP 三次握手的过程，仔细分析了 TCP 报头结构

不过虚拟机性能是真的差，码字都能卡住，而且 LiberOffice 图片难排版

我想在特别想学习 HTML，CSS，javascript 等搭建一个自己的网站，apache 的那个默认页面是可以更改的，可惜我还不会

这次实验攻击好像没有成功，其他的设备仍然可以正常访问？我猜测有如下几个原因：

- 现代操作系统有更强的资源管理能力，连接表更难溢出
- 攻击的频率不够快，都被处理掉了，真正的攻击都是每秒上万次
- 系统动态处理，当检测到大量 TCP SYN 请求时，会缩短 SYN timeout，以此防御攻击
- apache 自动采取了相应的防护措施，比如 Micro blocks, SYN cookies, RST cookies, Stack tweaking 等技术应对处理了 SYN flood 攻击

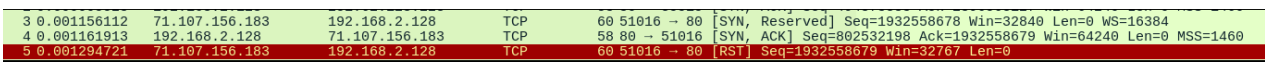
    Micro blocks，在服务器内存中为每个传入的 SYN 请求分配一个微记录（少至 16 个字节），而不是一个完整的 TCB

    SYN cookies，使用加密散列，服务器用一个序列号(seqno)发送它的 SYN-ACK 响应，该序列号由客户端 IP 地址、端口号以及可能的其他唯一标识信息构造而成。当客户端响应时，此哈希被包含在 ACK 包中。服务器验证 ACK，然后才为连接分配内存

    RST cookies， 对于来自给定客户端的第一个请求，服务器故意发送无效的 SYN-ACK。 这应该导致客户端生成一个 RST 数据包，它告诉服务器出现问题。 如果收到，服务器知道请求是合法的，记录客户端，并接受来自它的后续传入连接

    Stack tweaking， 管理员可以调整 TCP 堆栈以减轻 SYN 泛洪的影响，通过，有选择地丢弃传入连接 ，直到堆栈释放分配给连接的内存

疑问：如果两个虚拟机都是 NAT 模式时，奇怪的现象发生了



客户端会在两次握手后，立马发送 TCP RST 断开 TCP 连接，导致 SYN 攻击失败。我当时还在想是不是 python scapy 库自己干的，防止不法用途。

然后查了一下论坛，很多人说这是 linux 内核检测到异常 TCP 连接，自动终止了