**Socket Programming**

叶增渝 519030910168

1.拓扑结构解释

一个作为服务器的host h1，通过一个switch与所有的客户端host连接（明明分别为h2、h3、h4……）并传输数据

2.如何运行程序

（1）首先通过3个命令编译show.cc、client.cc与server.cc

gcc server.cc –o server –lpthread

gcc client.cc –o client

gcc show.cc –o show



其中server.cc为服务器host执行的源代码，将监听拓扑结构上的客户端请求，当连接时会创建一个子线程用于处理客户端文件请求由于多线程，需要参数-lpthread；client.cc为客户端client执行的源代码，用于接收文件并保存到本地文件中；show.cc的作用为显示后台执行程序打印的信息，由于我们需要多个host执行命令，所以需要多host同时执行任务，所以需要后端执行，但这不会在cmd中打印信息，需要一个default程序在前台执行完成打印信息。

（2）执行拓扑结构的构造

sudo python3 topo.py

C:\Users\HP\AppData\Local\Microsoft\Windows\INetCache\Content.Word\屏幕截图 2021-10-07 220913.png

输入需要产生的clients数量（小于等于20是由于server端监测的BUFFER\_SIZE大小仅为20），完成拓扑结构构造并进入Mininet CLI

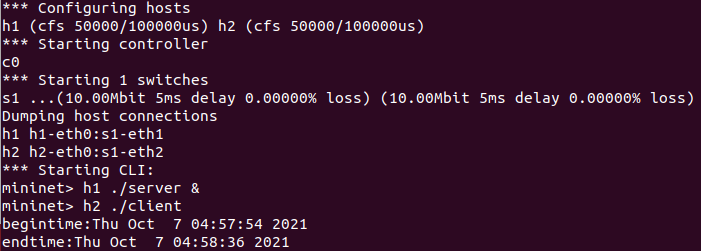
（3）指定不同的host完成不同的任务

指定h1执行server端任务（h1 ./server &），让剩下的host执行client程序（由于需要多host执行，都需要后台执行）（h2 ./client &），在完成传输后使用h2 ./show查看结果，计算时间

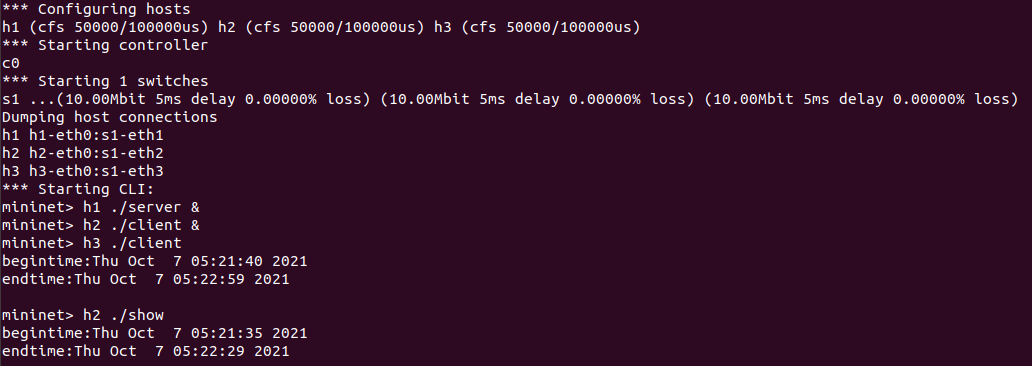
Tips:本地文件是根据拓扑结构中的名字命名的txt文件，server端保存着的是file.txt文件（一个50MB的文本文件），可以用gedit h2.txt命令打开查看

3.测试结果

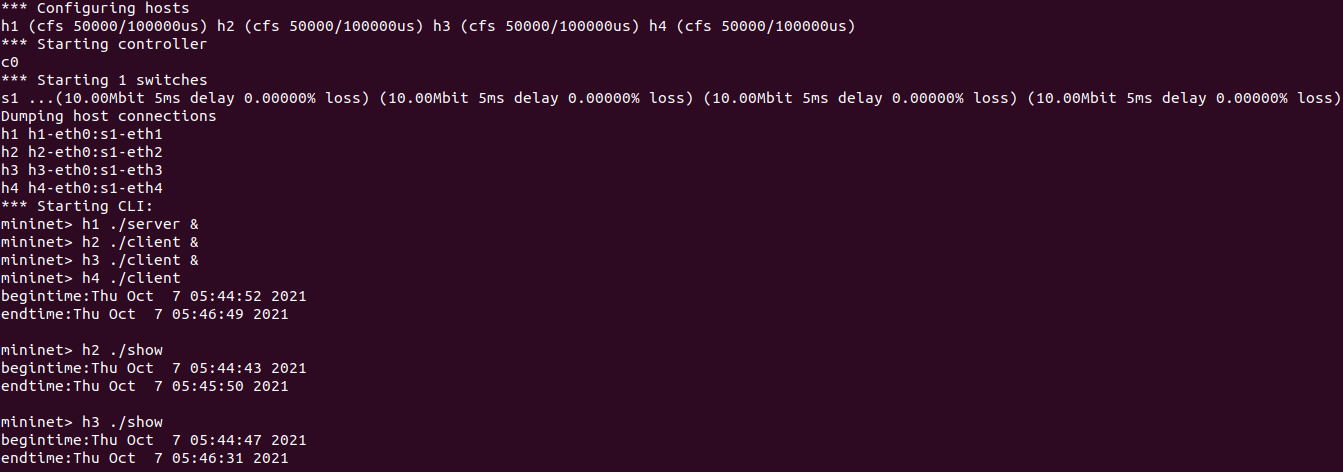
（1）测试截图



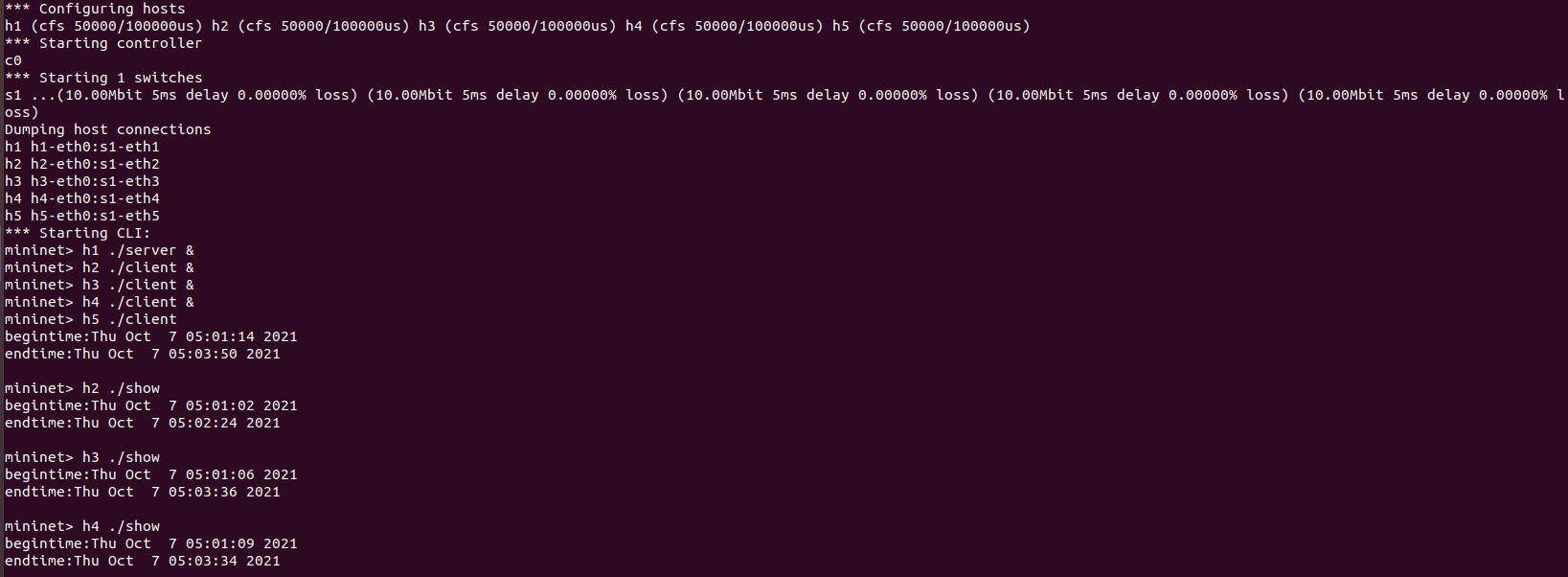
单client



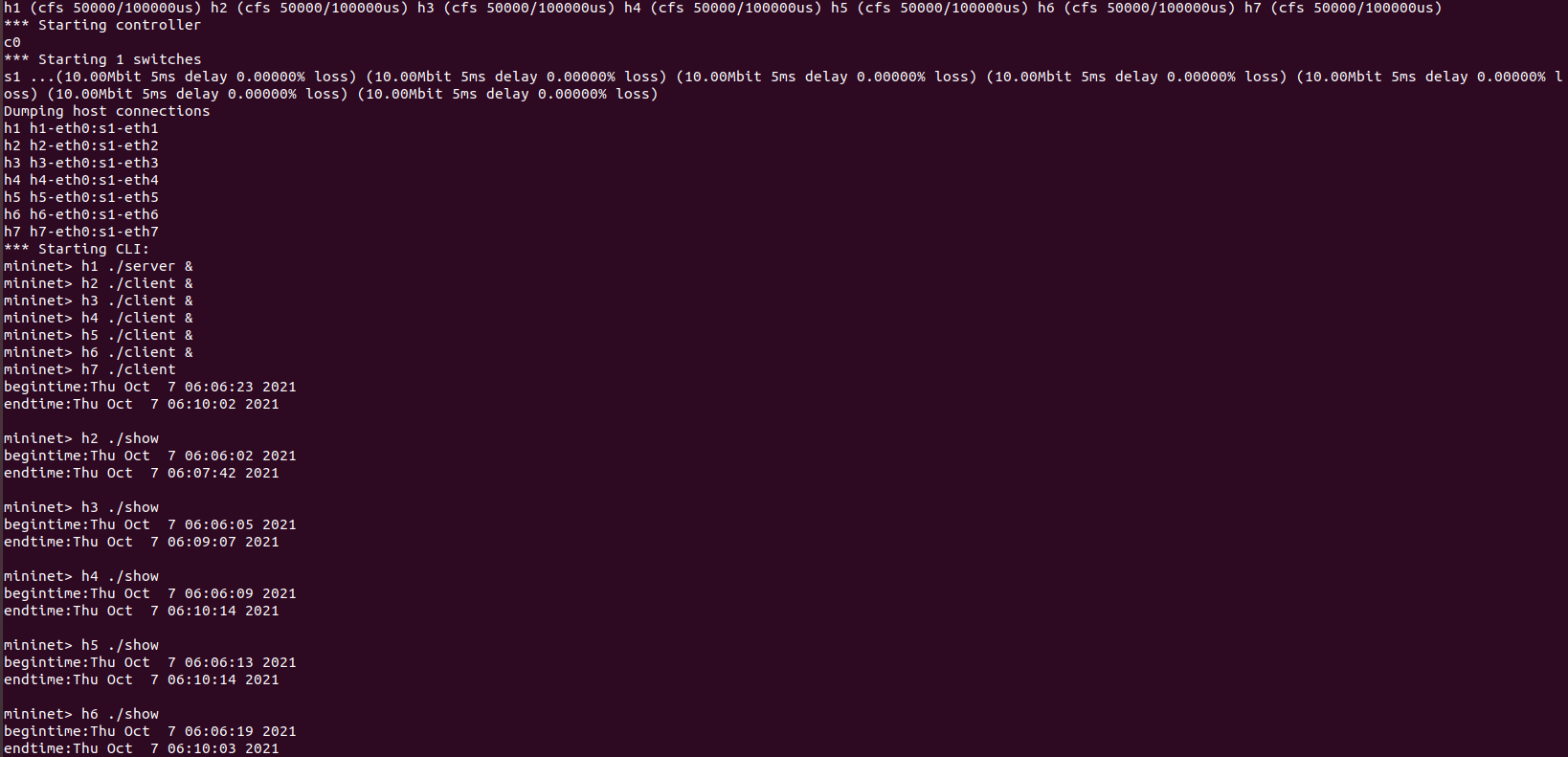
双clients



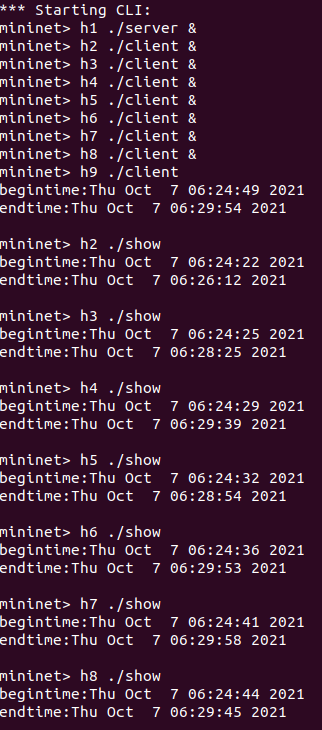
3 clients



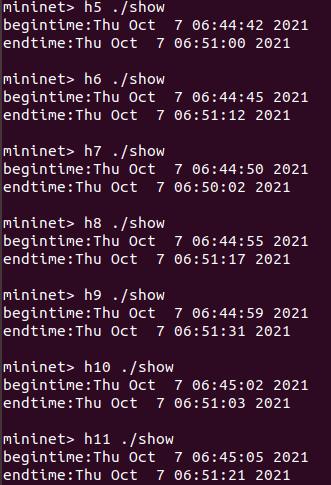
4 clients



6 clients



8 clients



10 clients

（2）结果汇总

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 客户端数量 | 1 | 2 | 3 | 4 | 6 | 8 | 10 |
| 开始时间 | 04：57：54 | 05：21：35 | 05：44：43 | 05：01：02 | 06：06：02 | 06：24：22 | 06：44：32 |
| 结束时间 | 04：58：36 | 05：22：59 | 05：46：49 | 05：03：50 | 06：10：14 | 06：29：58 | 06：51：21 |
| 总用时 | 42s | 84s | 126s | 168s | 252s | 336s | 409s |

（3）分析总结

当同时请求的服务端数量增加时，总共的下载时间将会增加。可能是由于总信道的带宽固定，即使可以多线程传输数据，总的传输量也有上限，在server-client模型中，客户端增加，总的下载量增加，时间也就增加了。