

Datacenter Network Simulation using ns3

April 29, 2018

姓名: 姜贵平
学号: SA17011142

1 实验要求

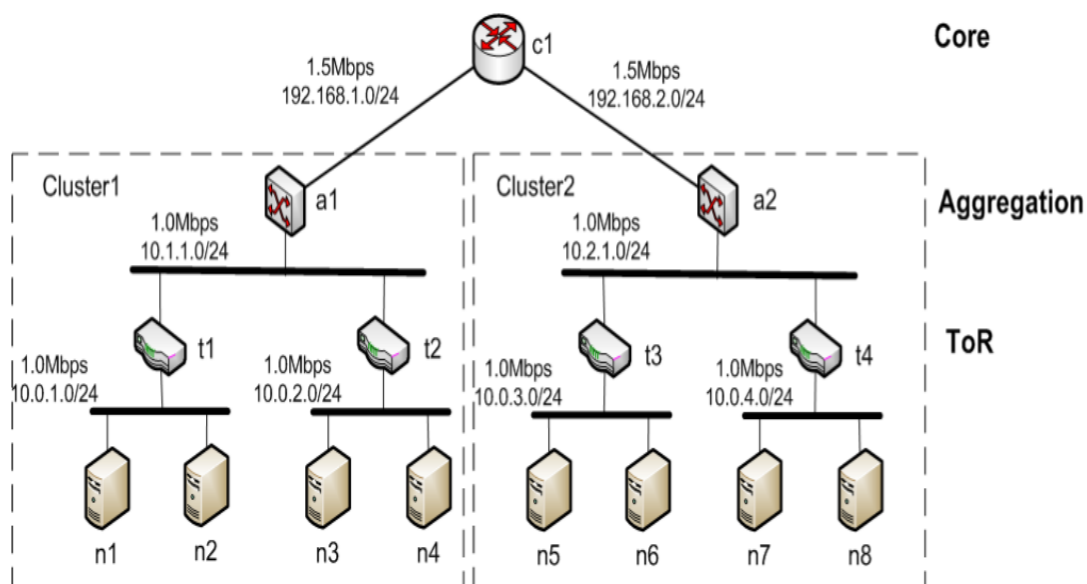


Figure 1: 网络拓扑结构

利用ns3编写程序模拟网络拓扑及网络消息传输状态（多对一和多对多），得到网络的吞吐量，找到网络瓶颈，提高网络带宽利用效率。

2 代码实现与

2.1 模式1: 多对多代码实现

2.1.1 创建节点

由网络结构可知, 网络需要创建15个节点, 包括:

- 8 servers
- 4 ToR switches
- 4 Aggregation switches
- 4 Core switch

其中 $c1, a1, a2$ 节点创建代码如下(其他节点类似):

Table 1: 创建节点

```
NodeContainer coragg1;  
coragg1.Create(2);  
NodeContainer coragg2;  
coragg2.Add(coragg1.Get(0));  
coragg2.Create(1);
```

2.1.2 设置网络属性

网络属性主要有两个, 1个是顶层连接的p2p网络和地下csma网络。设置网络属性, 并将其安装到相应节点上去。代码如下:

Table 2: 设置网络属性并安装

```
//Construct ptp topo upper  
PointToPointHelper ptp1,ptp2;  
sprintf(buf,"ptp1.SetDeviceAttribute ("DataRate", StringValue (buf));  
ptp1.SetChannelAttribute ("Delay", TimeValue (NanoSeconds (500)));  
ptp2.SetDeviceAttribute ("DataRate", StringValue (buf));  
ptp2.SetChannelAttribute ("Delay", TimeValue (NanoSeconds (500)));  
NetDeviceContainer devicePtp1,devicePtp2;  
devicePtp1=ptp1.Install (coragg1);  
devicePtp2=ptp2.Install (coragg2);  
//Construct csma topo  
CsmaHelper csma;  
sprintf(buf,"csma.SetChannelAttribute ("DataRate", StringValue (buf));  
csma.SetChannelAttribute ("Delay", TimeValue (NanoSeconds (500)));
```

2.1.3 安装协议栈分配ip

Table 3: 安装协议栈分配ip

```
stack.Install (coragg1);
stack.Install (coragg2.Get(1));
Ipv4AddressHelper address;
address.SetBase ("192.168.1.0", "255.255.255.0");
Ipv4InterfaceContainer interfacesPtp1 = address.Assign (devicePtp1);
address.SetBase ("192.168.2.0", "255.255.255.0");
Ipv4InterfaceContainer interfacesPtp2 = address.Assign (devicePtp2);
```

2.1.4 设置节点通信

设置appsink和client节点通信，每一个节点同时作为服务器和client，向另一个簇 中相应的节点发送和接受信息。举一个 $n1 < -- > n5$ 代码，其他类似。代码如下：

Table 4: 设置节点通信

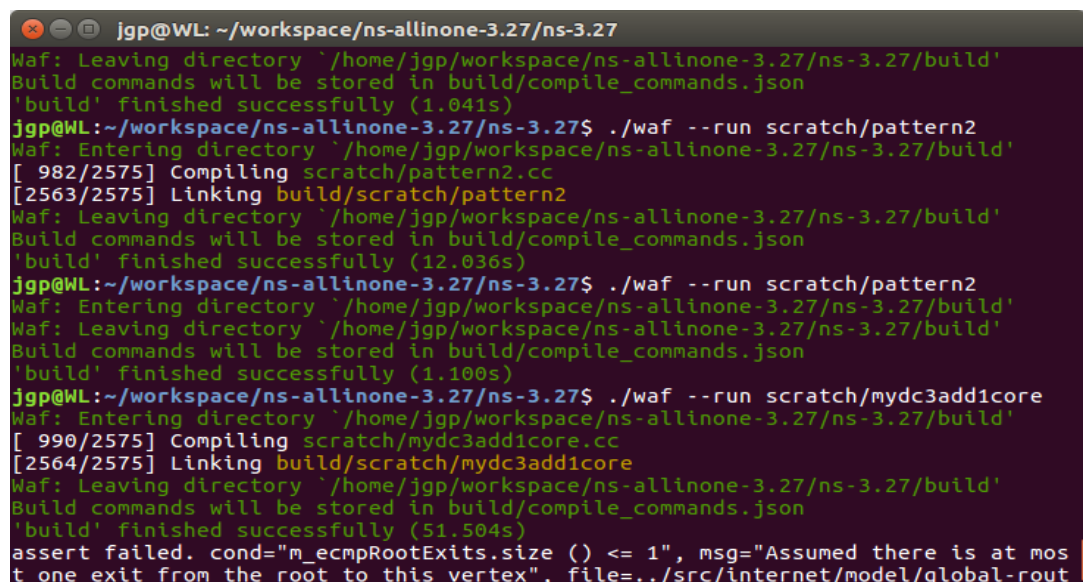
```
//Set up traffic n1 - n5
PacketSinkHelper packetSinkHelper1("ns3::TcpSocketFactory",
InetSocketAddress(interfacesCsma3.GetAddress(1),8080));
ApplicationContainer sinkApp1 =
packetSinkHelper1.Install(csmaNodes3.Get(1));
sinkApp1.Start(Seconds(0.0));
sinkApp1.Stop(Seconds(20.0));
OnOffHelper client1("ns3::TcpSocketFactory",
InetSocketAddress(interfacesCsma3.GetAddress(1), 8080));
client1.SetAttribute ("OnTime",
StringValue("ns3::ConstantRandomVariable[Constant=50]"));
client1.SetAttribute ("OffTime",
StringValue("ns3::ConstantRandomVariable[Constant=0]"));
client1.SetAttribute ("DataRate", DataRateValue
(DataRate("1.0Mbps")));
client1.SetAttribute ("PacketSize", UIntegerValue (2000));
ApplicationContainer clientApp1 = client1.Install (csmaNodes1.Get(1));
clientApp1.Start(Seconds (1.0 ));
clientApp1.Stop (Seconds (21.0));
```

2.1.5 利用pcap追踪网络，以后分析

Table 5: pcap追踪网络

```
csma.EnablePcap ("Pattern1 n8 to n4", deviceCsmaNodes2.Get (2), true);
ptp1.EnablePcapAll("Pattern1 ptp1");
ptp2.EnablePcapAll("Pattern1 ptp2");
ptp1.EnablePcap ("pattern1 c1",devicePtp1.Get (0), true);
ptp1.EnablePcap ("pattern1 a1",devicePtp1.Get (1), true);
ptp2.EnablePcap ("pattern1 a2",devicePtp2.Get (1), true);
csma.EnablePcap ("pattern1 t1",deviceAggToR1.Get (1), true);
csma.EnablePcap ("pattern1 t3",deviceAggToR2.Get (1), true);
```

2.1.6 实验运行



```
jgp@WL: ~/workspace/ns-allinone-3.27/ns-3.27
Waf: Leaving directory `/home/jgp/workspace/ns-allinone-3.27/ns-3.27/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (1.041s)
jgp@WL:~/workspace/ns-allinone-3.27/ns-3.27$ ./waf --run scratch/pattern2
Waf: Entering directory `/home/jgp/workspace/ns-allinone-3.27/ns-3.27/build'
[ 982/2575] Compiling scratch/pattern2.cc
[2563/2575] Linking build/scratch/pattern2
Waf: Leaving directory `/home/jgp/workspace/ns-allinone-3.27/ns-3.27/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (12.036s)
jgp@WL:~/workspace/ns-allinone-3.27/ns-3.27$ ./waf --run scratch/pattern2
Waf: Entering directory `/home/jgp/workspace/ns-allinone-3.27/ns-3.27/build'
Waf: Leaving directory `/home/jgp/workspace/ns-allinone-3.27/ns-3.27/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (1.100s)
jgp@WL:~/workspace/ns-allinone-3.27/ns-3.27$ ./waf --run scratch/mydc3add1core
Waf: Entering directory `/home/jgp/workspace/ns-allinone-3.27/ns-3.27/build'
[ 990/2575] Compiling scratch/mydc3add1core.cc
[2564/2575] Linking build/scratch/mydc3add1core
Waf: Leaving directory `/home/jgp/workspace/ns-allinone-3.27/ns-3.27/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (51.504s)
assert failed. cond="m_ecmpRootExits.size () <= 1", msg="Assumed there is at mos
t one exit from the root to this vertex", file=../src/internet/model/global-rout
```

Figure 2: 运行截图

2.2 模式2：一对多代码实现

实现过程与多对多类似，再设置节点通信事,将所有的节点除 n_1 外都与 n_1 通信即可。

3 实验分析与改进

3.1 工具

对生成的pcap文件利用wireshark工具进行分析。

3.2 模式一

3.2.1 实验分析

我们追踪每一个节点，产生如下pcap文件。

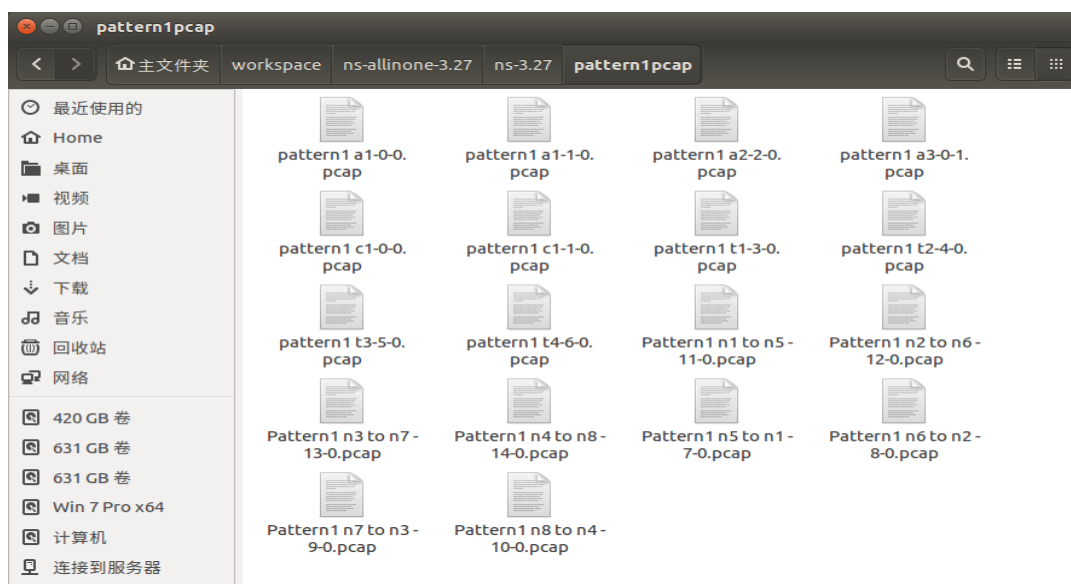


Figure 3: pcap追踪文件

由于pattern1网络对称性的结构，我们分析一下模式一下 n_1, t_1, a_1, c_1 所在节点的pcap文件。服务器 n_1 到 n_5 传输情况：



Figure 4: n_1 到 n_5 吞吐量

由图可知，只有41kb/s，远远小于最大发送1Mb/s,我们再看看传输情况。

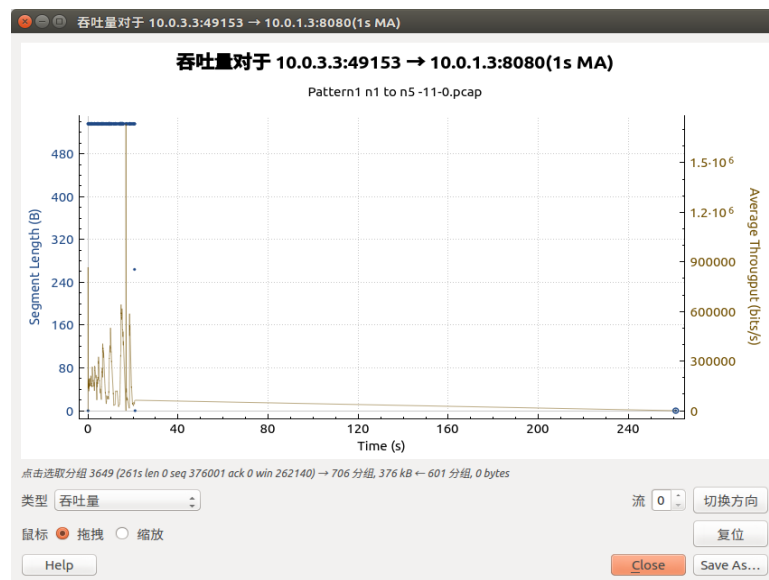


Figure 5: n_1 到 n_5 TCP传输情况

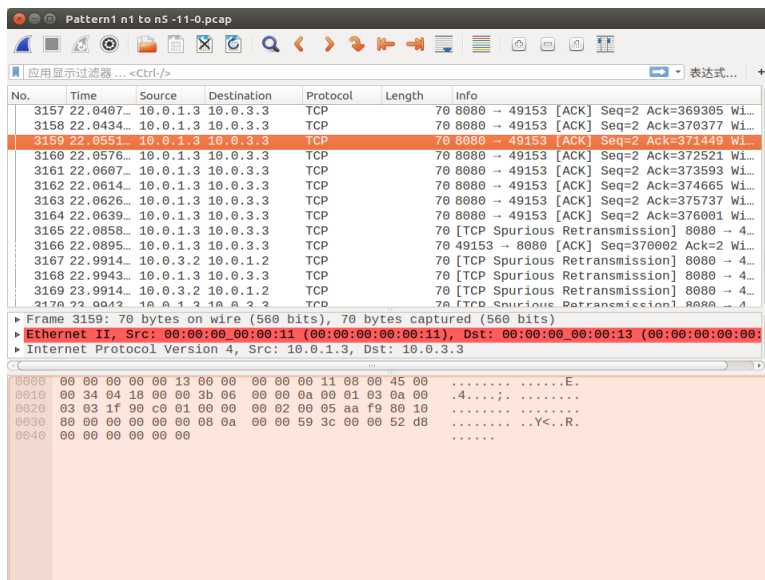


Figure 6: n_1 到 n_5 后期报文

我们通过观察上面两幅图，发现节点信息传输到后期基本为0，再看看报文，我们发现后期有大量确认信息ACK，重传严重。我们再看看交换机 t_1 的传输情况：



Figure 7: t_1 传输情况

由图可知，吞吐量为81kb/s,我们再看看传输情况。

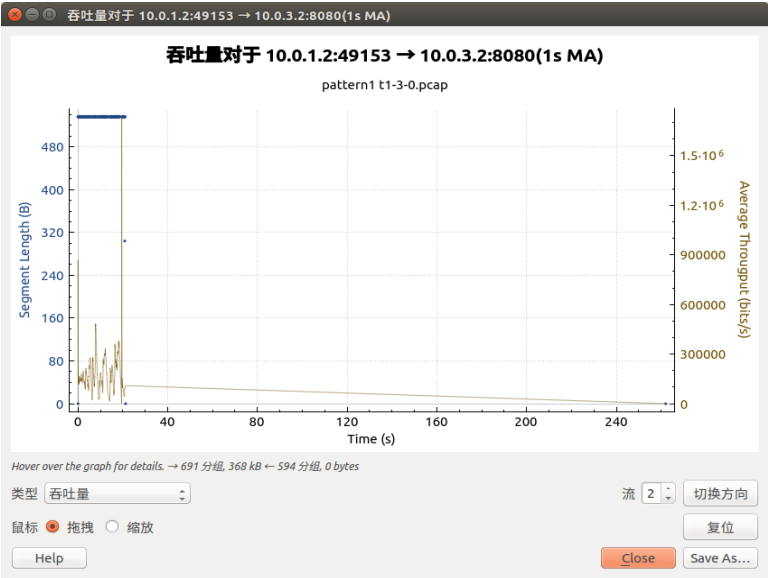


Figure 8: t_1 TCP传输情况

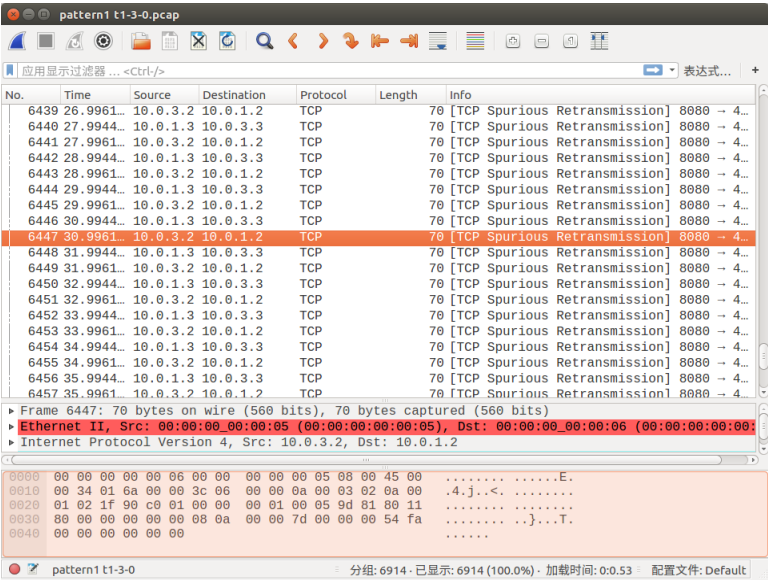


Figure 9: t_1 后期报文

我们对交换机 a_1 的进行上述分析，发现传输情况相似，至此我们可知是由

于8个节点同时当做服务器发送和接受，传输冲突太大，导致只在开始位置传输成功。

3.2.2 实验改进1

1.将服务器总体减少一半，采取如果一个节点只能作为服务器或者客户端，不能同时作为客户端和服务端。实验效果如下：



Figure 10: 改进后 n_1 吞吐量

由图可知，吞吐量为81kb/s,我们再看看传输情况。



Figure 11: 改进后 t_1 吞吐量

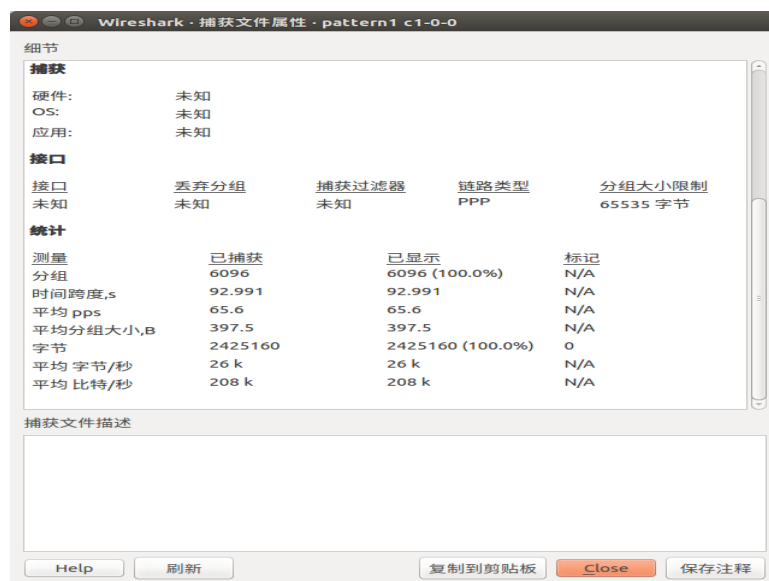


Figure 12: 改进后 c_1 吞吐量

我们发现传输明显改善，这验证了前面的观点，是由于服务器过多导致传输冲突严重导致的。

3.3 模式二

3.3.1 实验分析

我们将 n_1 作为唯一server，其他节点作为client发送数据到server上，我们追踪每一个节点，来发现模式二的瓶颈。我们先追踪 n_1 作为服务器的接受情况。如图：



Figure 13: 模式二下 n_1 传输情况

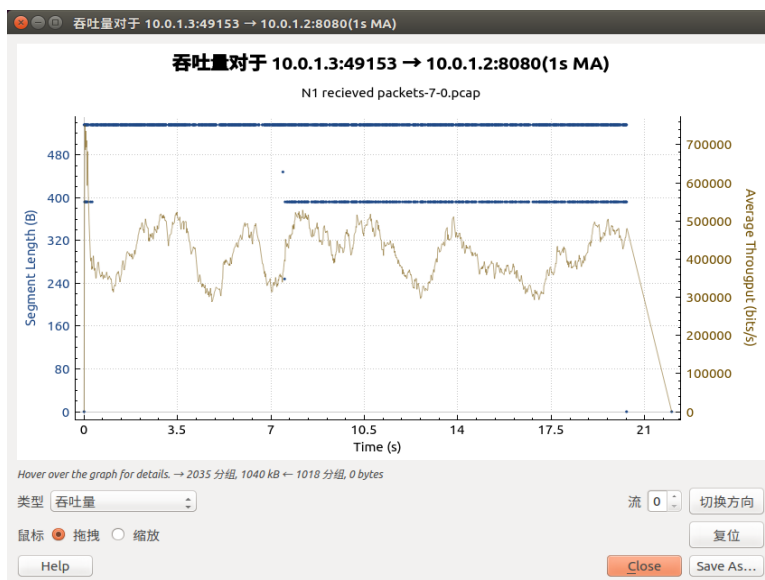


Figure 14: 模式二下 n_1 TCP传输情况

由图可知，吞吐量为973kb/s,基本已经占据了传输贷款上限。可以初步认定网络瓶颈就在最底层网络csma最大传输速率的限制下。我们看一下总体的网络带宽使用情况。绘制成下表：

	带宽(Mbps)	平军吞吐量
t_1	1	0.64
t_2	1	0.64
t_3	1	0.26
a_1	1	0.25
a_2	1	0.25
c_1	1.5	0.25
n_1	1	0.97

3.3.2 实验改进1

通过上表的分析，我们可以认定网络瓶颈就在底层csma网络最大传输速率上，我们增加底层网络传输带宽，增加到2.0Mbps。再看一下网络带宽使用情况。

	带宽(Mbps)	平均吞吐量
t_1	1	0.974
t_2	1	0.974
t_3	1	0.522
a_1	1	0.502
a_2	1	0.502
c_1	1.5	0.502
n_1	2	1.769

我们通过对上表分析，发现将最底层csma网络最大传输速率提高一倍，其他节点网络带宽几乎先行的提升了一倍，效果特别明显，中间网络已经快要接近瓶颈了。实验改进非常好。

4 实验总结

本次实验通过ns3模拟数据中心网络，通过利用pcap追踪节点传输情况，对两种不同模式下的网络进行了分析，并在拓扑以及通道传输进行了改进，提高了自己对底层网络传输的认识。