

# 负载均衡七层负载模式下下载速度降低典型案例分析

七层服务器负载均衡 孔凡安 2023-08-30 发表

## 组网及说明

组网简化如下：

客户端---LB---服务器

告警信息

bsj

#### 问题描述

四层负载场景下客户端下载速度相比于七层负载，速度下降了5倍

## 过程分析

现场场景比较简单，代理模式下将客户端的请求负载到内部服务器。

前期排查如下：

1. 代理模式下设备CPU、内存正常，无单核高情况；接口利用率处于较低水平。
2. 同样的测试条件下，将设备配置改为四层负载后，速度提升明显。

从以上的排查来看，分析可能是代理模式下设备处理上流程要更多，导致下载速率受到影响。

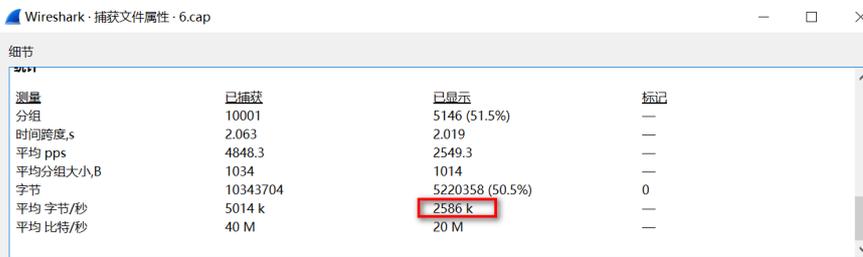
四层服务器负载均衡：可识别网络层和传输层信息，是基于流的负载均衡，通过对报文进行逐流分发，将同一条流的报文分发给同一台服务器。由于四层服务器负载均衡对七层业务无法按内容分发，从而限制了其适用范围。

七层服务器负载均衡：除了可识别网络层和传输层信息之外，还可识别应用层信息，是基于内容的负载均衡，通过对报文承载的内容进行深度解析，根据其中的内容进行逐包分发，按既定策略将连接导向指定的服务器，从而实现了业务范围更广泛的服务器负载均衡。因此七层负载对设备性能要求比较高，开启七层负载后，设备的吞吐量、新建、并发等参数均会受到影响。

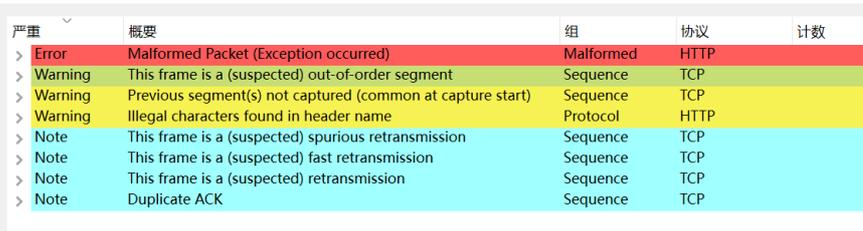
按照上述说法答复客户后，客户明显不接受这种说法，要求解决该问题。

遇到这种情况，那就抓个包看看吧。测试环境也不复杂，就是客户端从服务器上下载数据。于是我们在LB设备上抓取了代理前后的报文进行分析。

1. 从统计-捕获文件属性来看，平均速度为2.58 MB/s，和实际下载过程中速率差不多。

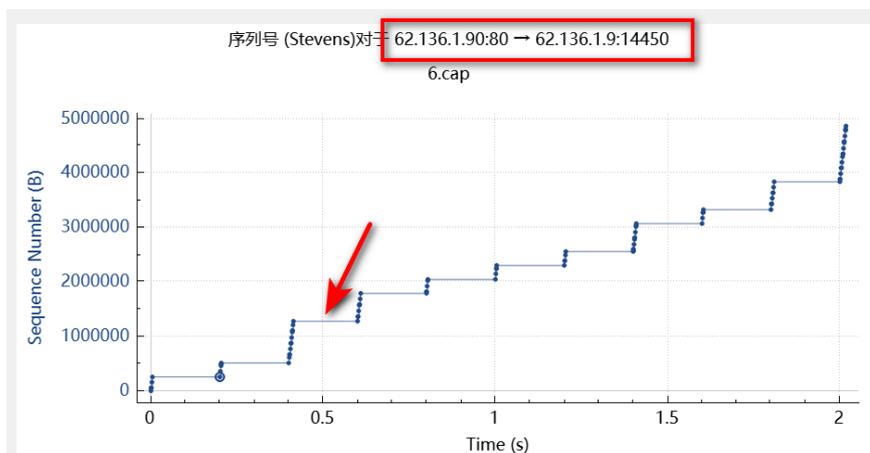


2. 从分析-专家信息的显示来看，也没有重传等问题，似乎也看不出来问题。



严重	概要	组	协议	计数
> Error	Malformed Packet (Exception occurred)	Malformed	HTTP	
> Warning	This frame is a (suspected) out-of-order segment	Sequence	TCP	
> Warning	Previous segment(s) not captured (common at capture start)	Sequence	TCP	
> Warning	Illegal characters found in header name	Protocol	HTTP	
> Note	This frame is a (suspected) spurious retransmission	Sequence	TCP	
> Note	This frame is a (suspected) fast retransmission	Sequence	TCP	
> Note	This frame is a (suspected) retransmission	Sequence	TCP	
> Note	Duplicate ACK	Sequence	TCP	

3. 通过统计-TCP流图形-时间序列查看，发现服务器 (ip.addr ==62.136.1.90) 回包给LB设备(ip.addr == 62.136.1.9)的时候存在时延，即发送下一个序列号报文的时候会花费较长时间。



定位到该报文位置可以发现服务器发送数据时时间差 (No.2611和No.2660) 大约在180 ms的时间，180 ms的时间对于下载速率的影响是十分巨大的。

Time	Source	Destination	Protocol	Length	Flags	Seq	Window	Len	Tsv
2099	2022-06-09 04:57:26.833406	62.136.1.90	62.136.1.90	TCP	128	0x2463 (9315)	1492	80 + 14450	[ACK] Seq=1276001 Ack=1 Win=8235 Len=1440 Tsv=1264666
2609	2022-06-09 04:57:26.833408	62.136.1.90	62.136.1.90	TCP	128	0x2464 (9316)	1492	80 + 14450	[ACK] Seq=1277441 Ack=1 Win=8235 Len=1440 Tsv=1264666
2611	2022-06-09 04:57:26.833427	62.136.1.90	62.136.1.90	TCP	128	0x2465 (9317)	1172	80 + 14450	[PSH, ACK] Seq=1278881 Ack=1 Win=8235 Len=1120 Tsv=132
2620	2022-06-09 04:57:26.833534	62.136.1.90	62.136.1.90	TCP	255	0xd082 (53346)	52	14450 + 80	[ACK] Seq=1 Ack=1247201 Win=65535 Len=0 Tsv=126527420
2623	2022-06-09 04:57:26.833585	62.136.1.90	62.136.1.90	TCP	255	0xd085 (53349)	52	14450 + 80	[ACK] Seq=1 Ack=1252961 Win=65535 Len=0 Tsv=126527420
2628	2022-06-09 04:57:26.833636	62.136.1.90	62.136.1.90	TCP	255	0xd088 (53352)	52	14450 + 80	[ACK] Seq=1 Ack=125841 Win=65535 Len=0 Tsv=126527420
2633	2022-06-09 04:57:26.833687	62.136.1.90	62.136.1.90	TCP	255	0xd08b (53355)	52	14450 + 80	[ACK] Seq=1 Ack=126391 Win=65535 Len=0 Tsv=126527420
2636	2022-06-09 04:57:26.833737	62.136.1.90	62.136.1.90	TCP	255	0xd08e (53358)	52	14450 + 80	[ACK] Seq=1 Ack=126941 Win=65535 Len=0 Tsv=126527420
2639	2022-06-09 04:57:26.833787	62.136.1.90	62.136.1.90	TCP	255	0xd091 (53361)	52	14450 + 80	[ACK] Seq=1 Ack=127491 Win=65535 Len=0 Tsv=126527420
2642	2022-06-09 04:57:26.833837	62.136.1.90	62.136.1.90	TCP	255	0xd094 (53364)	52	14450 + 80	[ACK] Seq=1 Ack=128041 Win=65535 Len=0 Tsv=126527420
2647	2022-06-09 04:57:26.833887	62.136.1.90	62.136.1.90	TCP	255	0xd097 (53367)	52	14450 + 80	[ACK] Seq=1 Ack=128591 Win=65535 Len=0 Tsv=126527420
2650	2022-06-09 04:57:26.833937	62.136.1.90	62.136.1.90	TCP	255	0xd09a (53370)	52	14450 + 80	[ACK] Seq=1 Ack=129141 Win=65535 Len=0 Tsv=126527420
2655	2022-06-09 04:57:26.834008	62.136.1.90	62.136.1.90	TCP	255	0xd09d (53373)	52	14450 + 80	[ACK] Seq=1 Ack=129691 Win=65535 Len=0 Tsv=126527420
2659	2022-06-09 04:57:27.019433	62.136.1.90	62.136.1.90	TCP	255	0xd09f (53375)	52	14450 + 80	[ACK] Seq=1 Ack=1280001 Win=65535 Len=0 Tsv=126527439
2660	2022-06-09 04:57:27.019675	62.136.1.90	62.136.1.90	TCP	128	0x246a (9388)	1492	80 + 14450	[ACK] Seq=1280001 Ack=1 Win=8235 Len=1440 Tsv=1264666
2662	2022-06-09 04:57:27.019680	62.136.1.90	62.136.1.90	TCP	128	0x246b (9389)	1492	80 + 14450	[ACK] Seq=1281441 Ack=1 Win=8235 Len=1440 Tsv=1264666

那么是什么原因导致服务器发送数据的时候有这个延迟呢，我们继续分析抓包可以看到No.2611 (Seq+Len) 的确认报文是No.2659(Ack)。比较两个报文的时间差发现延迟大约就是180 ms，这也就意味着是LB的ack确认报文延迟发送了。从现象来看似乎是服务器要等到LB设备的ACK确认后才会发送下一个报文。

Time	Source	Destination	Protocol	Length	Flags	Seq	Window	Len	Tsv
2608	2022-06-09 04:57:26.833406	62.136.1.90	62.136.1.90	TCP	128	0x2463 (9315)	1492	80 + 14450	[ACK] Seq=1276001 Ack=1 Win=8235 Len=1440 Tsv=1264666
2609	2022-06-09 04:57:26.833408	62.136.1.90	62.136.1.90	TCP	128	0x2464 (9316)	1492	80 + 14450	[ACK] Seq=1277441 Ack=1 Win=8235 Len=1440 Tsv=1264666
2611	2022-06-09 04:57:26.833423	62.136.1.90	62.136.1.90	TCP	128	0x2466 (9318)	1172	80 + 14450	[PSH, ACK] Seq=1278881 Ack=1 Win=8235 Len=1120 Tsv=132
2612	2022-06-09 04:57:26.833427	62.136.1.90	62.136.1.90	TCP	255	0xd05c (53340)	52	14450 + 80	[ACK] Seq=1 Ack=1247201 Win=65535 Len=0 Tsv=126527420
2615	2022-06-09 04:57:26.833479	62.136.1.90	62.136.1.90	TCP	255	0xd05f (53343)	52	14450 + 80	[ACK] Seq=1 Ack=1252961 Win=65535 Len=0 Tsv=126527420
2620	2022-06-09 04:57:26.833534	62.136.1.90	62.136.1.90	TCP	255	0xd062 (53346)	52	14450 + 80	[ACK] Seq=1 Ack=125841 Win=65535 Len=0 Tsv=126527420
2623	2022-06-09 04:57:26.833585	62.136.1.90	62.136.1.90	TCP	255	0xd065 (53349)	52	14450 + 80	[ACK] Seq=1 Ack=126391 Win=65535 Len=0 Tsv=126527420
2628	2022-06-09 04:57:26.833636	62.136.1.90	62.136.1.90	TCP	255	0xd068 (53352)	52	14450 + 80	[ACK] Seq=1 Ack=126941 Win=65535 Len=0 Tsv=126527420
2631	2022-06-09 04:57:26.833687	62.136.1.90	62.136.1.90	TCP	255	0xd06b (53355)	52	14450 + 80	[ACK] Seq=1 Ack=127491 Win=65535 Len=0 Tsv=126527420
2636	2022-06-09 04:57:26.833737	62.136.1.90	62.136.1.90	TCP	255	0xd06e (53358)	52	14450 + 80	[ACK] Seq=1 Ack=128041 Win=65535 Len=0 Tsv=126527420
2639	2022-06-09 04:57:26.833787	62.136.1.90	62.136.1.90	TCP	255	0xd071 (53361)	52	14450 + 80	[ACK] Seq=1 Ack=128591 Win=65535 Len=0 Tsv=126527420
2642	2022-06-09 04:57:26.833837	62.136.1.90	62.136.1.90	TCP	255	0xd074 (53364)	52	14450 + 80	[ACK] Seq=1 Ack=129141 Win=65535 Len=0 Tsv=126527420
2647	2022-06-09 04:57:26.833905	62.136.1.90	62.136.1.90	TCP	255	0xd077 (53367)	52	14450 + 80	[ACK] Seq=1 Ack=129691 Win=65535 Len=0 Tsv=126527420
2650	2022-06-09 04:57:26.833955	62.136.1.90	62.136.1.90	TCP	255	0xd07a (53370)	52	14450 + 80	[ACK] Seq=1 Ack=130241 Win=65535 Len=0 Tsv=126527420
2655	2022-06-09 04:57:26.834008	62.136.1.90	62.136.1.90	TCP	255	0xd07d (53373)	52	14450 + 80	[ACK] Seq=1 Ack=1278881 Win=65535 Len=0 Tsv=126527420
2659	2022-06-09 04:57:27.019433	62.136.1.90	62.136.1.90	TCP	255	0xd07f (53375)	52	14450 + 80	[ACK] Seq=1 Ack=1280001 Win=65535 Len=0 Tsv=126527439
2660	2022-06-09 04:57:27.019675	62.136.1.90	62.136.1.90	TCP	128	0x246a (9388)	1492	80 + 14450	[ACK] Seq=1280001 Ack=1 Win=8235 Len=1440 Tsv=1264666
2661	2022-06-09 04:57:27.019676	62.136.1.90	62.136.1.90	TCP	128	0x246b (9389)	1492	80 + 14450	[ACK] Seq=1281441 Ack=1 Win=8235 Len=1440 Tsv=1264666
2662	2022-06-09 04:57:27.019680	62.136.1.90	62.136.1.90	TCP	128	0x246c (9390)	1492	80 + 14450	[ACK] Seq=1282881 Ack=1 Win=8235 Len=1440 Tsv=1264666

```

Transmission Control Protocol, Src Port: 80, Dst Port: 14450, Seq: 1278881, Ack: 1, Len: 1120
Source Port: 80
Destination Port: 14450
[Stream index: 1]
[Conversation completeness: Incomplete (12)]
[TCP Segment Len: 1120]
Sequence Number: 1278881 (relative sequence number)
Sequence Number: 1278881 (relative sequence number)
Next Sequence Number: 1280001 (relative sequence number)
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 1509233186
1000 ---- = header length: 32 bytes (8)

```

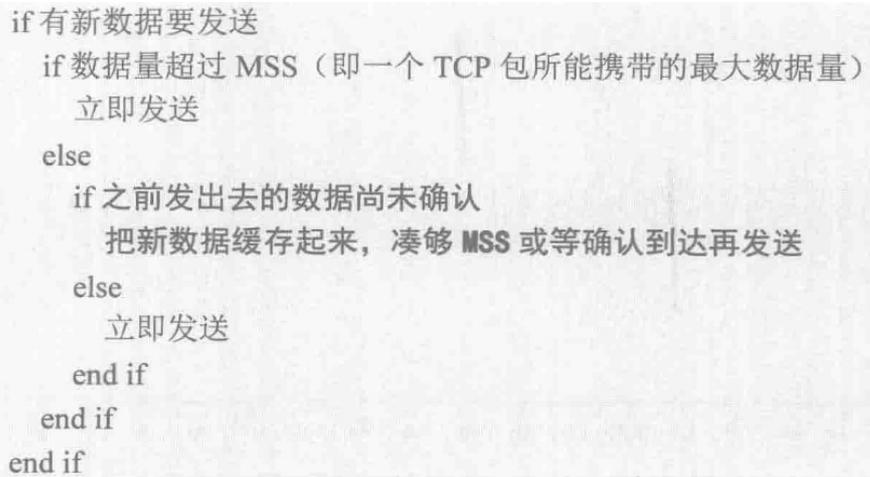
理论上问题分析到这里就结束了，优化一下LB就行啦~

但是可以思考一下，为什么前面服务器发送的报文LB没有出现延迟应答的情况呢？那么抓包可以给你答案，关注No.2608、No.2609以及No.2655报文发现，LB是每两个TCP分段确认一次 (LB.Ack==Server的Seq+Len)，如下：

Time	Source	Destination	Protocol	Length	Flags	Seq	Window	Len	Tsv
2601	2022-06-09 04:57:26.833324	62.136.1.90	62.136.1.90	TCP	128	0x2460 (9312)	1492	80 + 14450	[ACK] Seq=1273121 Ack=1 Win=8235 Len=1440 Tsv=1264666
2602	2022-06-09 04:57:26.833326	62.136.1.90	62.136.1.90	TCP	128	0x2461 (9313)	1492	80 + 14450	[ACK] Seq=1274561 Ack=1 Win=8235 Len=1440 Tsv=1264666
2605	2022-06-09 04:57:26.833406	62.136.1.90	62.136.1.90	TCP	255	0xd059 (53337)	52	14450 + 80	[ACK] Seq=1 Ack=1242701 Win=65535 Len=0 Tsv=126527420
2608	2022-06-09 04:57:26.833406	62.136.1.90	62.136.1.90	TCP	128	0x2463 (9315)	1492	80 + 14450	[ACK] Seq=1276001 Ack=1 Win=8235 Len=1440 Tsv=1264666
2609	2022-06-09 04:57:26.833408	62.136.1.90	62.136.1.90	TCP	128	0x2464 (9316)	1492	80 + 14450	[ACK] Seq=1277441 Ack=1 Win=8235 Len=1440 Tsv=1264666
2611	2022-06-09 04:57:26.833423	62.136.1.90	62.136.1.90	TCP	128	0x2466 (9318)	1172	80 + 14450	[PSH, ACK] Seq=1278881 Ack=1 Win=8235 Len=1120 Tsv=132
2612	2022-06-09 04:57:26.833534	62.136.1.90	62.136.1.90	TCP	255	0xd05c (53340)	52	14450 + 80	[ACK] Seq=1 Ack=1247201 Win=65535 Len=0 Tsv=126527420
2615	2022-06-09 04:57:26.833479	62.136.1.90	62.136.1.90	TCP	255	0xd05f (53343)	52	14450 + 80	[ACK] Seq=1 Ack=1252961 Win=65535 Len=0 Tsv=126527420
2620	2022-06-09 04:57:26.833534	62.136.1.90	62.136.1.90	TCP	255	0xd062 (53346)	52	14450 + 80	[ACK] Seq=1 Ack=125841 Win=65535 Len=0 Tsv=126527420
2623	2022-06-09 04:57:26.833585	62.136.1.90	62.136.1.90	TCP	255	0xd065 (53349)	52	14450 + 80	[ACK] Seq=1 Ack=126391 Win=65535 Len=0 Tsv=126527420
2628	2022-06-09 04:57:26.833636	62.136.1.90	62.136.1.90	TCP	255	0xd068 (53352)	52	14450 + 80	[ACK] Seq=1 Ack=126941 Win=65535 Len=0 Tsv=126527420
2631	2022-06-09 04:57:26.833687	62.136.1.90	62.136.1.90	TCP	255	0xd06b (53355)	52	14450 + 80	[ACK] Seq=1 Ack=127491 Win=65535 Len=0 Tsv=126527420
2636	2022-06-09 04:57:26.833737	62.136.1.90	62.136.1.90	TCP	255	0xd06e (53358)	52	14450 + 80	[ACK] Seq=1 Ack=128041 Win=65535 Len=0 Tsv=126527420
2639	2022-06-09 04:57:26.833787	62.136.1.90	62.136.1.90	TCP	255	0xd071 (53361)	52	14450 + 80	[ACK] Seq=1 Ack=128591 Win=65535 Len=0 Tsv=126527420
2642	2022-06-09 04:57:26.833837	62.136.1.90	62.136.1.90	TCP	255	0xd074 (53364)	52	14450 + 80	[ACK] Seq=1 Ack=129141 Win=65535 Len=0 Tsv=126527420
2647	2022-06-09 04:57:26.833905	62.136.1.90	62.136.1.90	TCP	255	0xd077 (53367)	52	14450 + 80	[ACK] Seq=1 Ack=129691 Win=65535 Len=0 Tsv=126527420
2650	2022-06-09 04:57:26.833955	62.136.1.90	62.136.1.90	TCP	255	0xd07a (53370)	52	14450 + 80	[ACK] Seq=1 Ack=130241 Win=65535 Len=0 Tsv=126527420
2655	2022-06-09 04:57:26.834008	62.136.1.90	62.136.1.90	TCP	255	0xd07d (53373)	52	14450 + 80	[ACK] Seq=1 Ack=1278881 Win=65535 Len=0 Tsv=126527420

而前面提到的No.2611报文正好是TCP的最后一个分段，猜测LB设备应该是要等一个定时器超时时间 (TCP delay ack机制) 才会确认。

那么基于以上的分析，我们大概可以复盘整个问题，应用系统或存储系统TCP传输应该采用了Nagle算法，有兴趣的可以百度一下。可以简单理解为如果要发送的数据量小于MSS，要等到对方的ACK确认后才会继续发送数据。下面这个图也比较形象：



那么Nagle算法和LB延迟确认的机制 (7层SLB目前实现是TCP delay ack，延迟应答。即200毫秒定时器内收到两个TCP分段，发送TCP ACK；若只收到1个TCP分段，定时器超时后发送TCP ACK) 碰撞到一起，就出现了现场这种情况。

