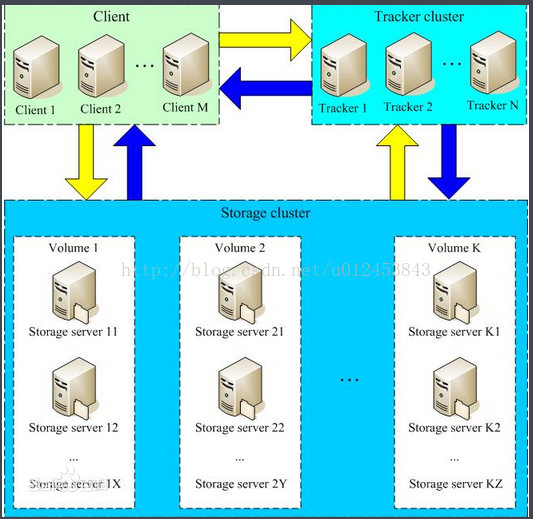
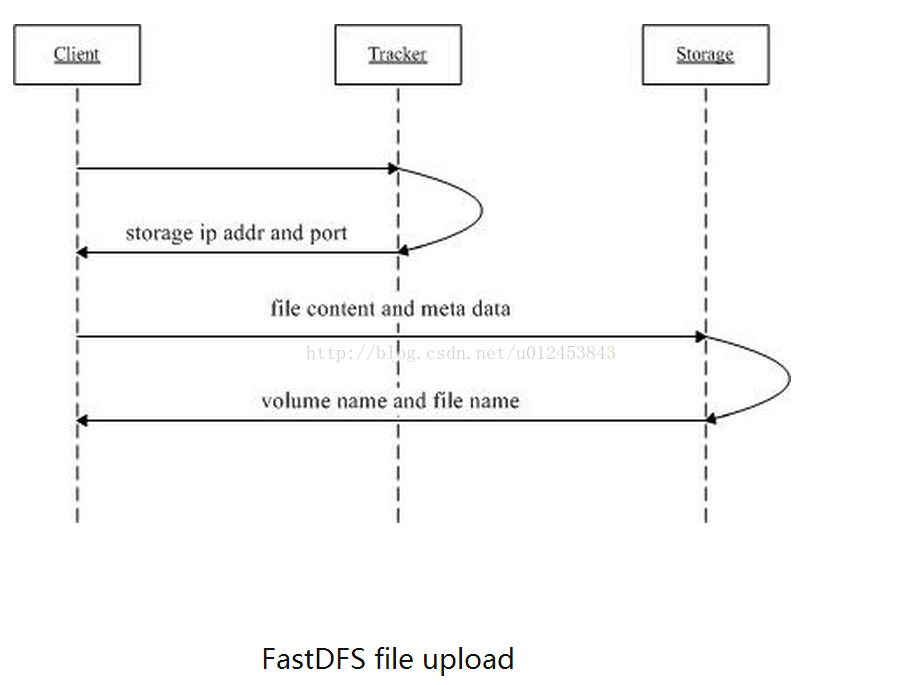
   FastDFS是一个开源的轻量级分布式文件系统，它对文件进行管理，功能包括：文件存储、文件同步、文件访问（文件上传、文件下载）等，解决了大容量存储和负载均衡的问题。特别适合以文件为载体的在线服务，如相册网站、视频网站等等。

       FastDFS为互联网量身定制，充分考虑了冗余备份、负载均衡、线性扩容等机制，并注重高可用、高性能等指标，使用FastDFS很容易搭建一套高性能的文件服务器集群提供文件上传、下载等服务。

       下面来看一张官网提供的集群图片，如下图所示。可以看到，Client端可以有多个，也就是同时支持多个客户端对FastDFS集群服务进行访问，Tracker是跟踪器，负责协调Client与Storage之间的交互，为了实现高可用性，需要用多个Tracker来做为跟踪器。Storage是专门用来存储东西的，而且是分组进行存储的，每一组可以有多台设备，这几台设备存储的内容完全一致，这样做也是为了高可用性，当现有分组容量不够时，我们可以水平扩容，即增加分组来达到扩容的目的。另外需要注意的一点是，如果一组中的设备容量大小不一致，比如设备A容量是80G，设备B的容量是100G，那么这两台设备所在的组的容量会以小的容量为准，也就是说，当存储的东西大小超过80G时，我们将无法存储到该组中了。Client端在与Storage进行交互的时候也与Tracker cluster进行交互，说的通俗点就是Storage向Tracker cluster进行汇报登记，告诉Tracker现在自己哪些位置还空闲，剩余空间是多大。



 下面再来看一张上传图片的时序图，从中可以看到，Client想上传图片，它先向Tracker进行询问，Tracker查看一下登记信息之后，告诉Client哪个storage当前空闲，Tracker会把IP和端口号都返回给Client，Client在拿到IP和端口号之后，便不再需要通过Tracker，直接便向Storage进行上传图片，Storage在保存图片的同时，会向Tracker进行汇报，告诉Tracker它当前是否还留有剩余空间，以及剩余空间大小。汇报完之后，Storage将服务器上存储图片的地址返回给Client，Client可以拿着这个地址进行访问图片。

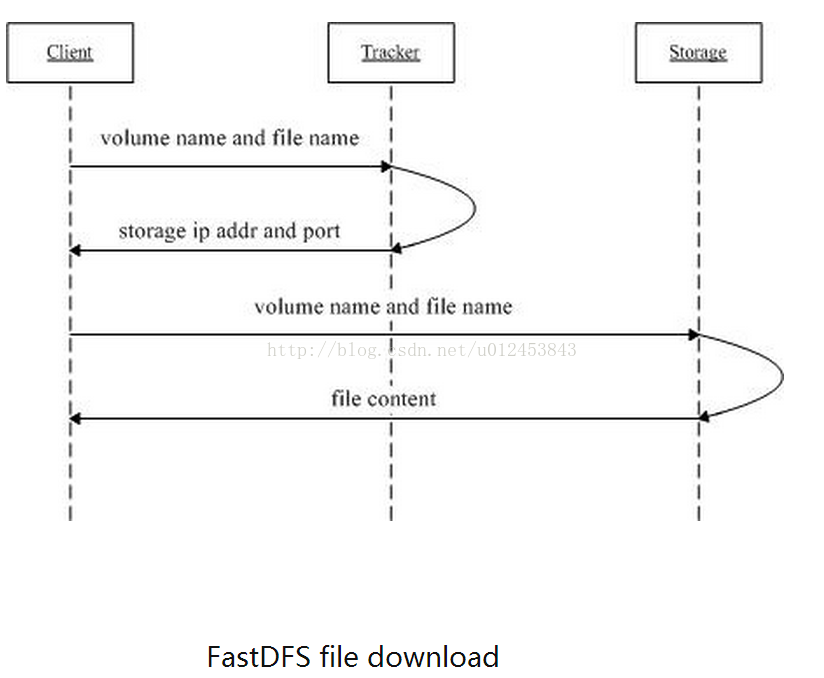


再看一张下载图片的时序图，步骤如下：

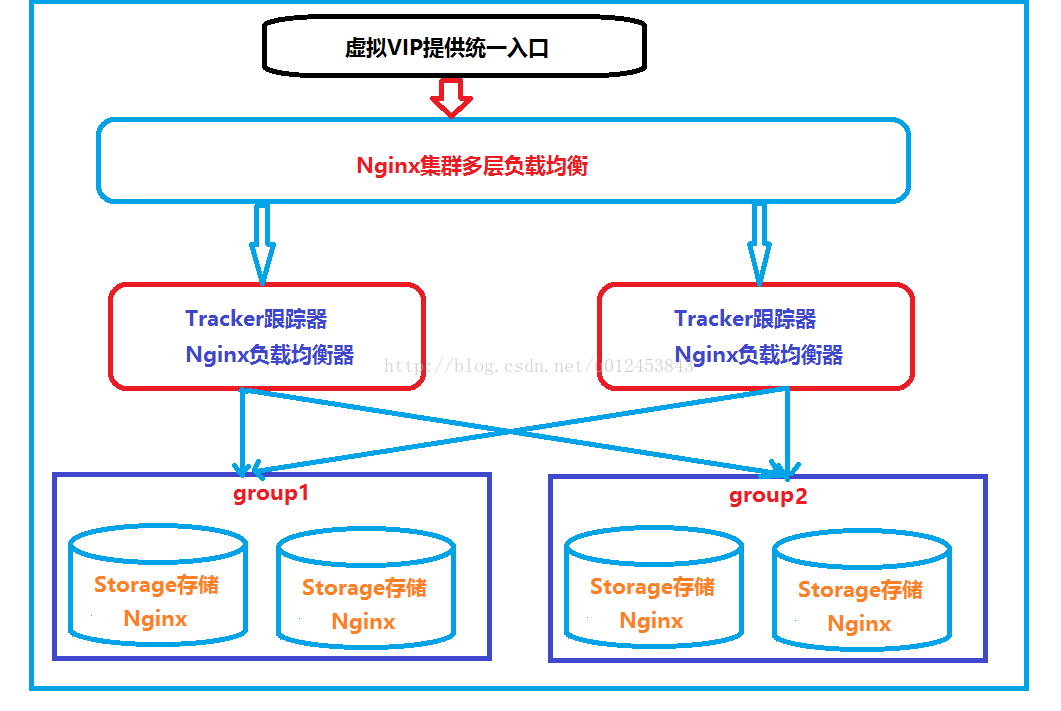
1. client询问tracker下载文件的storage，参数为文件标识（卷名和文件名）；

2. tracker返回一台可用的storage；

3. client直接和storage通讯完成文件下载。



下面我们便来搭建一个FastDFS集群，我们搭建一个如下图所示的集群，总共需要8台虚拟机

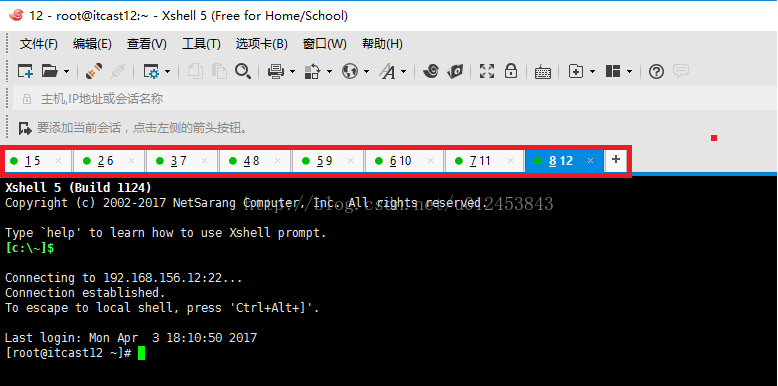


  操作步骤如下：

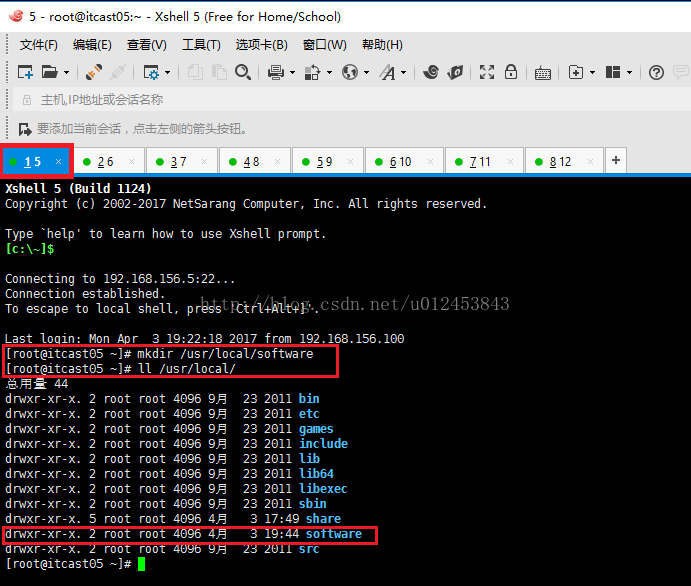
1.安装8台虚拟机（最小化安装，大家可以参考：http://blog.csdn.net/u012453843/article/details/68947589这篇博客进行安装）

2.给这8台虚拟机配置静态IP并且要能上网，大家可以参考：http://blog.csdn.net/u012453843/article/details/52839105这篇博客进行配置，不过由于现在是最小化安装，是没有安装vim命令的，因此需要使用"vi"命令来修改文件。

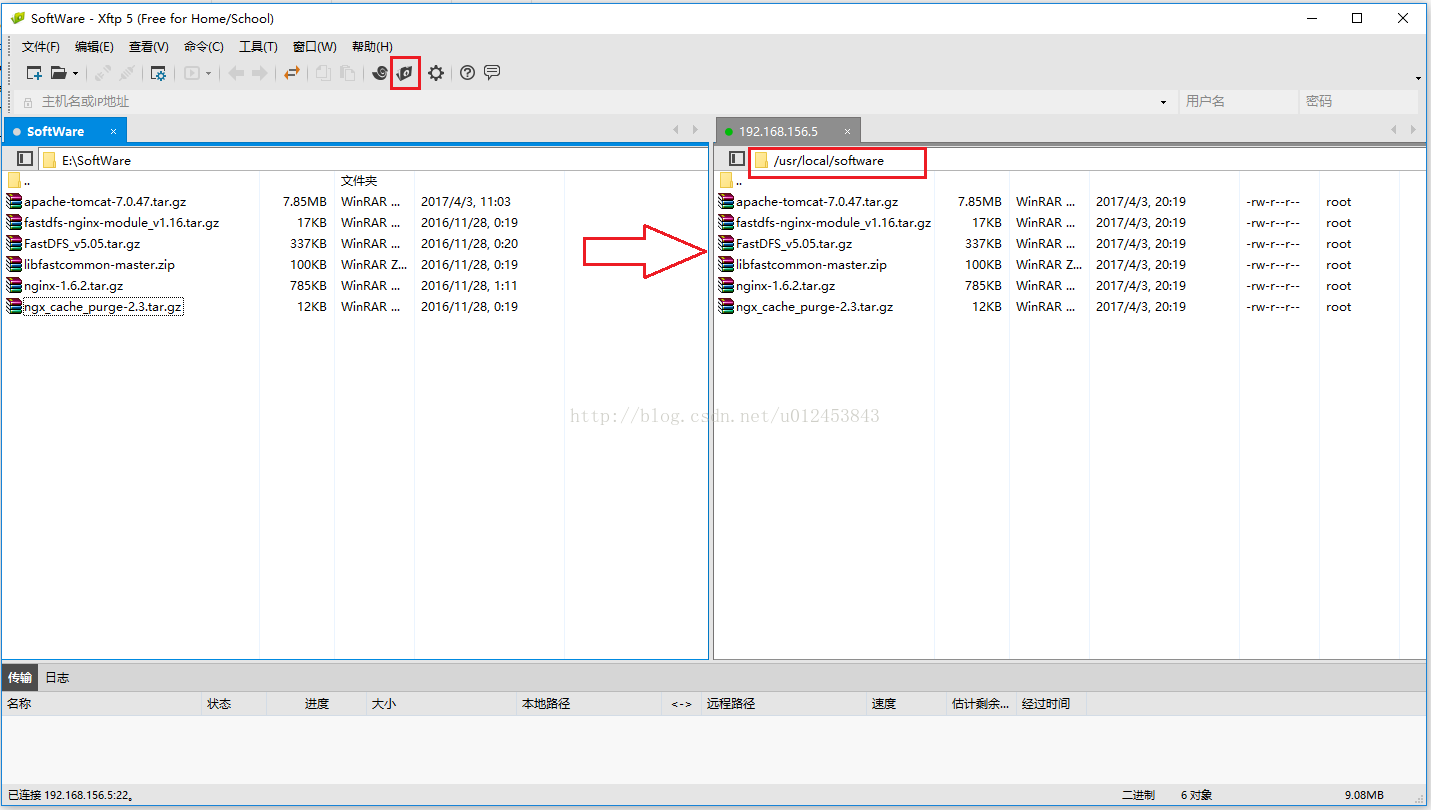
3.配置好静态IP之后，我们使用XShell工具来操作虚拟机（因为真实环境中我们是不大可能直接去操作服务器的，都是通过远程连接工具来进行操作的）。如下图所示，我使用的虚拟机分别是192.168.156.5、192.168.156.6、192.168.156.7、192.168.156.8、192.168.156.9、192.168.156.10、192.168.156.11、192.168.156.12。其中，192.168.156.5、192.168.156.6分别作为tracker1和tracker2。192.168.156.7、192.168.156.8作为group1，192.168.156.9、192.168.156.10作为group2。我把192.168.156.11、192.168.156.12作为Nginx集群多层负载均衡。多层负载均衡会生成一个虚拟IP，我们最终会通过虚拟IP来访问我们的集群。我们先安装tracker和group的六台设备。



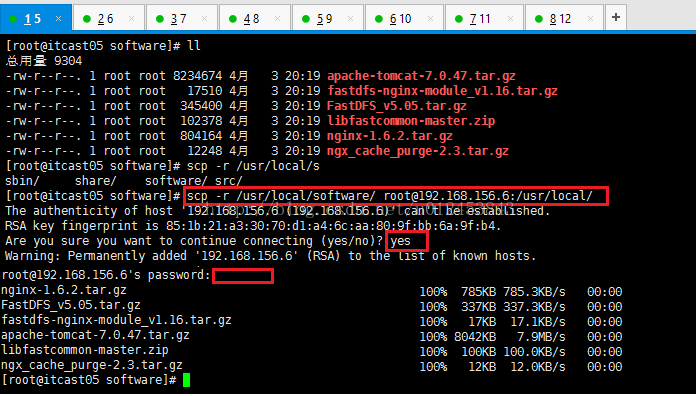
4.在192.168.156.5的"/usr/local/"目录下新建一个"software"目录，用来存放我们的安装文件。这样做的好处是容易管理。如下图所示



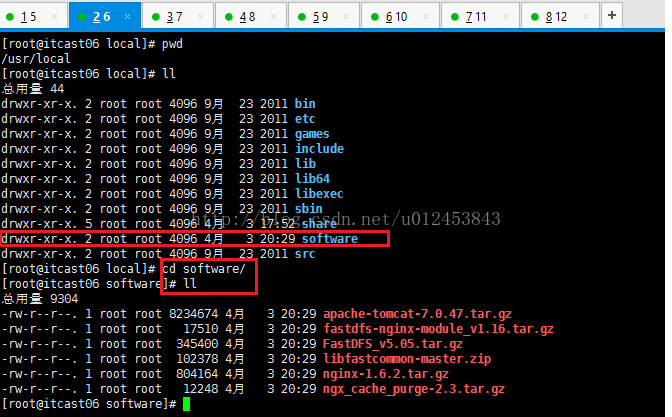
5.我们一次性上传我们所需要的所有安装文件，大家可以到：这个地址进行下载。我们使用XShell结合Xftp5进行上传，关于如何结合大家可以参考：http://blog.csdn.net/u012453843/article/details/68951776这篇文件进行操作。如下图所示。



6.我们将192.168.156.5这台设备上刚上传的文件复制到其它设备上。我们使用的命令是：scp -r /usr/local/software/ root@192.168.156.6:/usr/local/，这句命令的意思是，使用scp的方式将softWare及其下的文件都复制到192.168.156.6的/usr/local目录下，如果192.168.156.6的/usr/local目录下没有softWare目录，那么会自动创建这么一个目录。"root@"的意思是指定传到哪个用户组下面。由于当前都是操作的root用户，因此也可以不用写"root@"。输入命令并按回车后，会让我们输入是否继续，我们输入"yes"并回车，之后会让我们输入192.168.156.6的root用户的密码，我们输入之后便开始上传操作了（如果输入scp命令后很久才能到提示 让我们输入yes/no，那么我们可以在/etc/hosts文件中配置下所有要参与互相通信的ip和名称的映射关系），如下图所示。



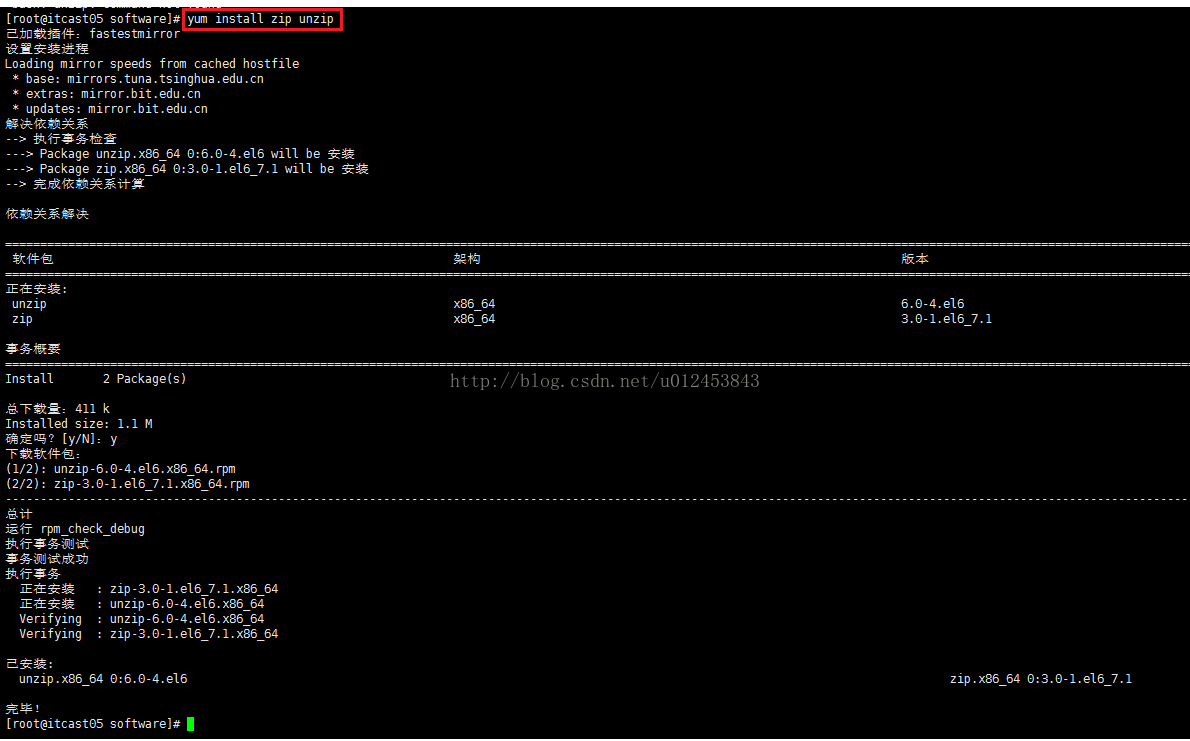
上传完之后，我们到192.168.156.6的/usr/local目录下查看一下，发现自动多了softWare目录。 我们再进入softWare目录内，发现确实已经复制过来了。同理，我们再向192.168.156.6、192.168.156.7、192.168.156.8、192.168.156.9、192.168.156.10这几台虚拟机复制这些安装文件。



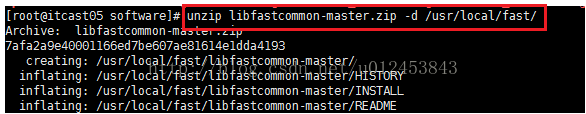
7.安装gcc，使用的命令：yum install make cmake gcc gcc-c++。在192.168.156.6、192.168.156.7、192.168.156.8、192.168.156.9、192.168.156.10虚拟机也都安装下gcc环境。



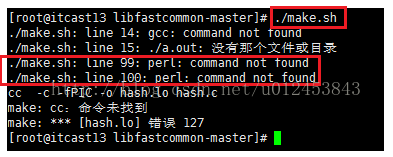
8.最小化安装是没有zip和unzip命令的，所以需要安装，安装命令：yum install zip unzip。同样，为192.168.156.6、192.168.156.7、192.168.156.8、192.168.156.9、192.168.156.10也安装zip和unzip命令。



9.解压libfastcommon-master.zip。使用命令：unzip libfastcommon-master.zip -d /usr/local/fast/。解压到/usr/local/fast/是为了便于管理。同理解压192.168.156.6、192.168.156.7、192.168.156.8、192.168.156.9、192.168.156.10虚拟机中的该文件。



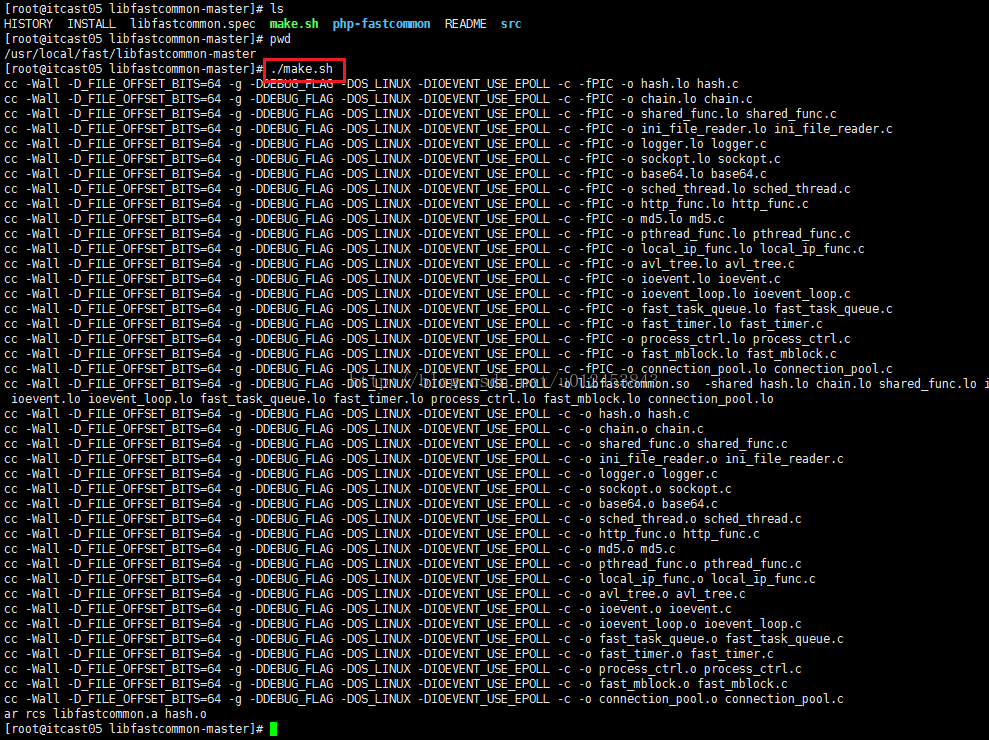
10.安装vim，之所以安装vim是因为安装vim会自动帮我们安装perl，否则我们使用./make.sh来编译的时候会报错，如下图所示。



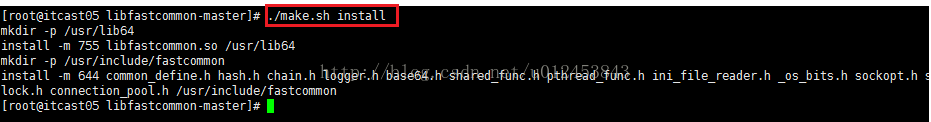
安装vim所使用命令：yum install vim-enhanced。同理在192.168.156.6、192.168.156.7、192.168.156.8、192.168.156.9、192.168.156.10虚拟机也安装vim命令。



11.编译libfastcommon-master，我们到/usr/local/fast/libfastcommon-master目录下，使用命令：./make.sh来进行编译。

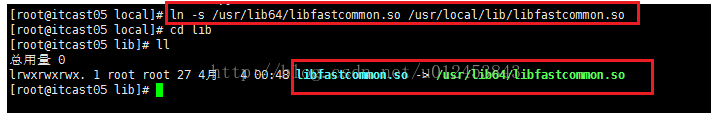


 接着执行./make.sh install命令。如下图所示。（我们把./make.sh和./make.sh install命令在192.168.156.6、192.168.156.7、192.168.156.8、192.168.156.9、192.168.156.10虚拟机的相同目录下也执行一遍）



12.创建软链接，我们从上图中可以看到执行./make.sh install命令后把fastcommon安装在了/usr/lib64下面，这个目录不是我们常用的目录，如果不实用软链接的话，将来安装的东西多了，它们的安装目录我们将比较难找到，不便于管理操作，为了在我们熟悉的目录下看到我们安装的目录，我们使用软链接来实现。

       创建第一条软链接的命令：ln -s /usr/lib64/libfastcommon.so /usr/local/lib/libfastcommon.so，如下图所示。



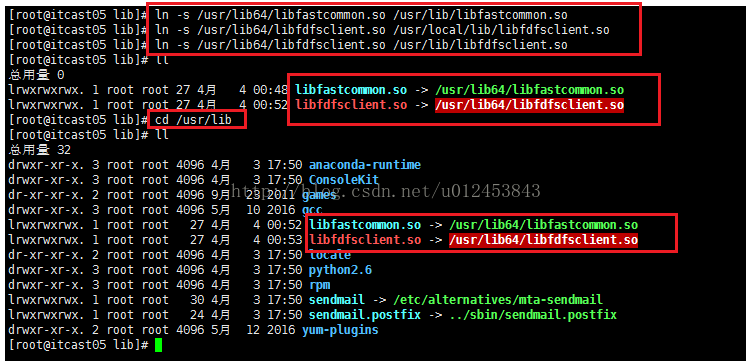
接着再创建三条软链接，命令依次如下：

[root@itcast05 lib]# ln -s /usr/lib64/libfastcommon.so /usr/lib/libfastcommon.so

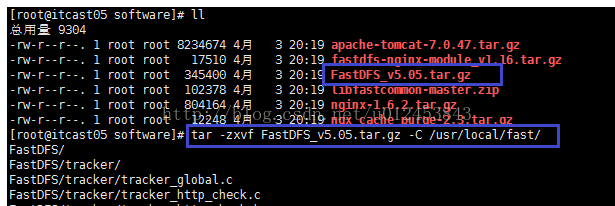
[root@itcast05 lib]# ln -s /usr/lib64/libfdfsclient.so /usr/local/lib/libfdfsclient.so

[root@itcast05 lib]# ln -s /usr/lib64/libfdfsclient.so /usr/lib/libfdfsclient.so

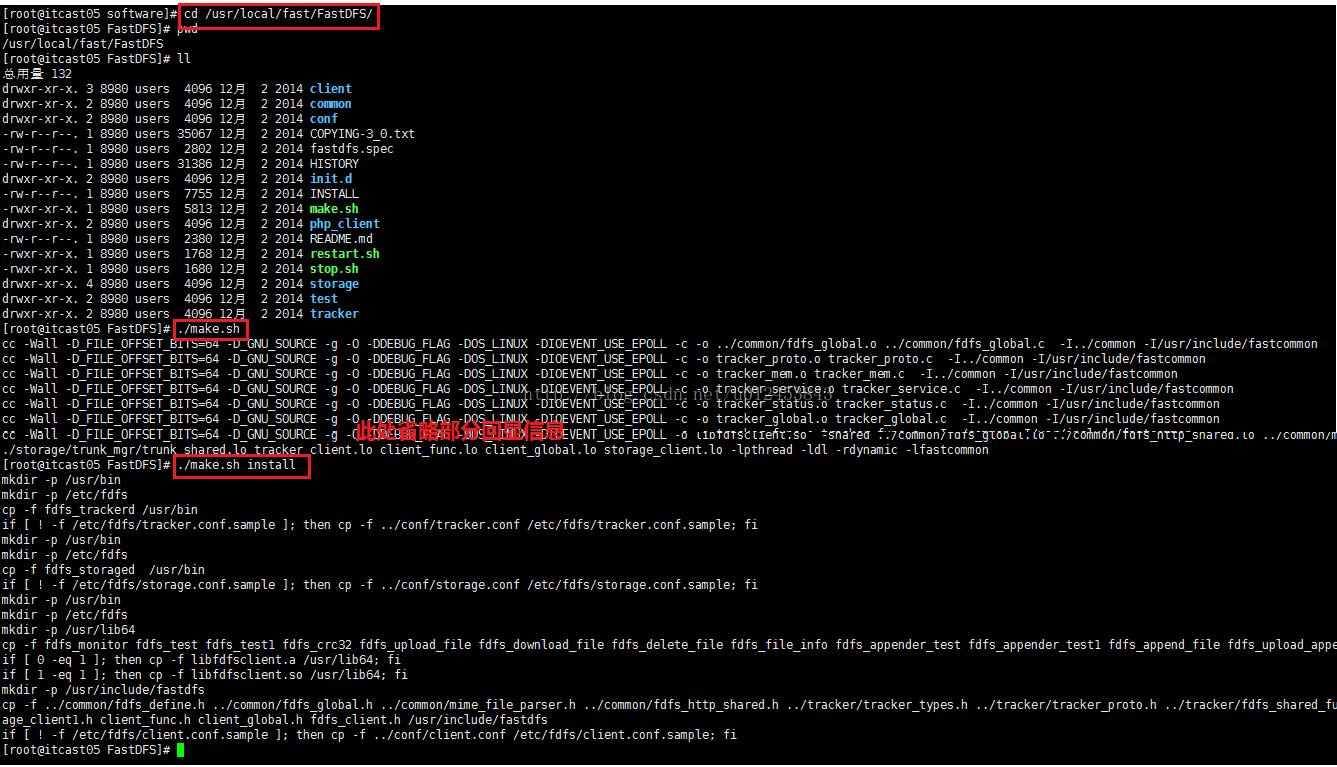
      创建完软链接之后，我们再查看软链接，如下图所示，发现有一条软链接是红色的，一闪一闪的，这是由于fastclient还未安装造成的，随着后续的安装，这个报警会自动消失。同理，我们在192.168.156.6、192.168.156.7、192.168.156.8、192.168.156.9、192.168.156.10上也创建同样的软链接。



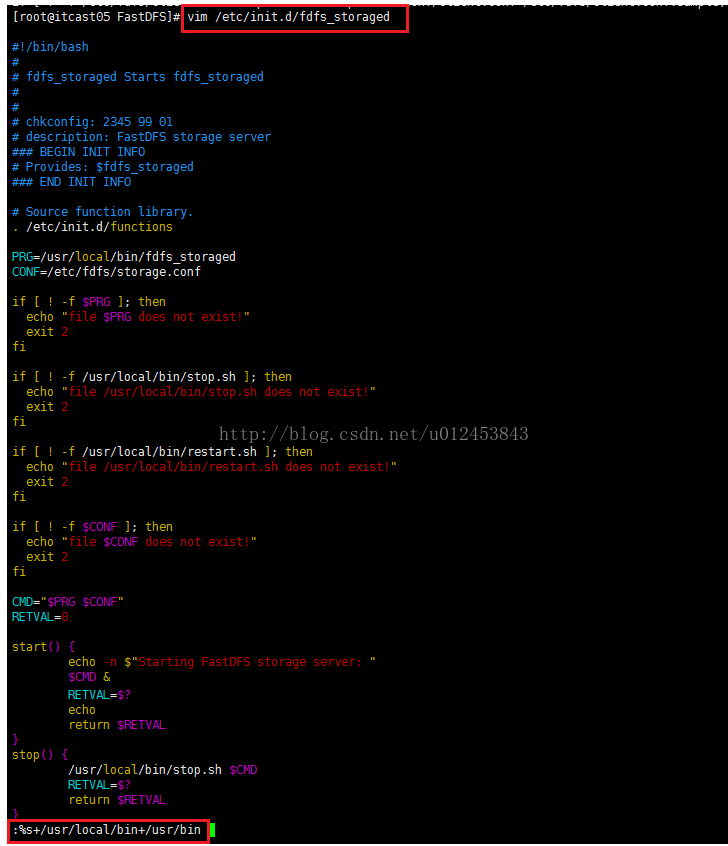
13.安装FastDFS，我们先到/usr/local/softWare/目录下，然后使用命令：tar -zxvf FastDFS\_v5.05.tar.gz -C /usr/local/fast/进行解压，如下图所示。



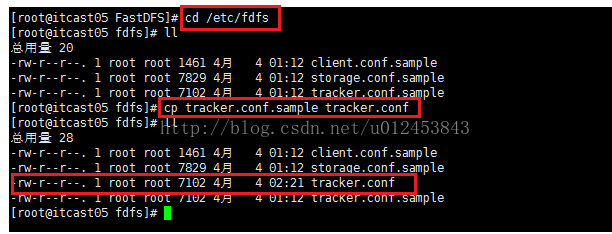
 解压完后，我们进入到/usr/local/fast/fastDFS/目录下，依次执行./make.sh和./make.sh install命令进行安装。如下图所示，同理，我们在192.168.156.6、192.168.156.7、192.168.156.8、192.168.156.9、192.168.156.10上也进行与第13步同样的安装操作。



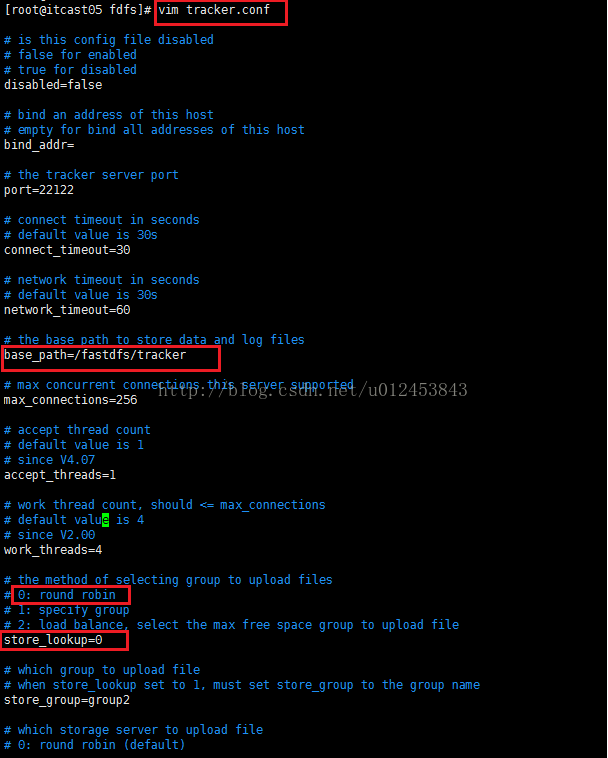
14.因为FastDFS服务脚本设置的bin目录为/usr/local/bin/下,但是实际我们安装在了/usr/bin/下面。所以我们需要修改FastDFS配置文件中的路径，也就是需要修改俩 个配置文件： 命令：vim /etc/init.d/fdfs\_storaged 输入一个":"，然后输入全局替换命令：%s+/usr/local/bin+/usr/bin并按回车即可完成替换，替换完之后，保存退出该文件，然后再打开看一下是否都已经将/usr/local/bin替换成/usr/bin了。同样的步骤，输入第二条命令：vim /etc/init.d/fdfs\_trackerd 进行全局替换，替换命令：%s+/usr/local/bin+/usr/bin。同样为192.168.156.6、192.168.156.7、192.168.156.8、192.168.156.9、192.168.156.10进行替换操作。



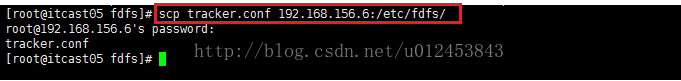
15.前面做的都是公共部分的配置，下面我们来配置tracker1(192.168.156.5)和tracker2(192.168.156.6)。我们先到/etc/fdfs/目录下，使用cp tracker.conf.sample tracker.conf复制一份tracker.conf.sample并更名为tracker.conf。



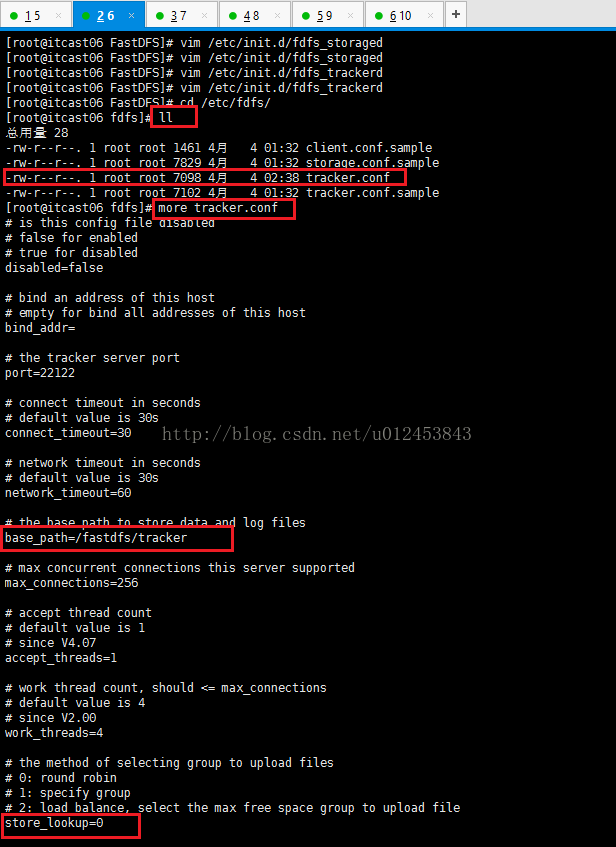
  编辑tracker.conf文件，需要修改的地方有两处，第一处：base\_path，将默认的路径修改为/fastdfs/tracker。第二处：store\_lookup，该值默认是2（即负载均衡策略），现在把它修改为0（即轮询策略，修改成这样方便一会儿我们进行测试，当然，最终还是要改回到2的。如果值为1的话表明要始终向某个group进行上传下载操作，这时下图中的"store\_group=group2"才会起作用，如果值是0或2，则"store\_group=group2"不起作用）。



   由于192.168.156.6和192.168.156.5的tracker配置是一样的，因此我们只需要将192.168.156.5上配置好的tracker.conf文件复制一份到192.168.156.6上。使用的命令：scp tracker.conf 192.168.156.6:/etc/fdfs/



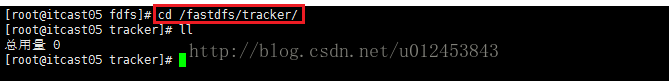
复制完之后，我们到192.168.156.6的/etc/fdfs/目录下查看一下是否已经有tracker.conf文件了，如下图所示，我们发现已经有该文件并且配置完全一样。



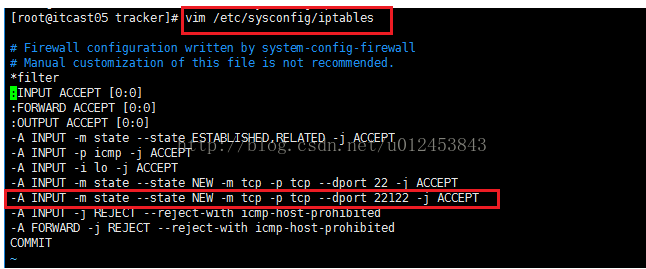
由于我们给base\_path配置的路径/fastdfs/tracker当前并不存在，因此我们需要在192.168.156.5和192.168.156.6上创建一下该目录，创建命令：mkdir -p /fastdfs/tracker，其中-p表示递归创建目录。

https://img-blog.csdn.net/20170403175125253?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvdTAxMjQ1Mzg0Mw==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center

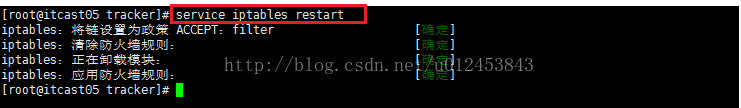
配置完了tracker1和tracker2，现在我们来启动两个tracker。我们先到我们刚创建的/fastdfs/tracker/目录下，发现当前该目录下什么也没有，如下图所示。



   在启动前，我们需要先在192.168.156.5和192.168.156.6这两台设备上配置一下防火墙，添加端口22122，从而可以让其它设备可以访问22122端口。添加的内容：-A INPUT -m state --state NEW -m tcp -p tcp --dport 22122 -j ACCEPT，如下图所示（注意所有的例子都是以一个为例，其它照着操作就可以了）。



添加完之后，我们重启防火墙，如下图所示。



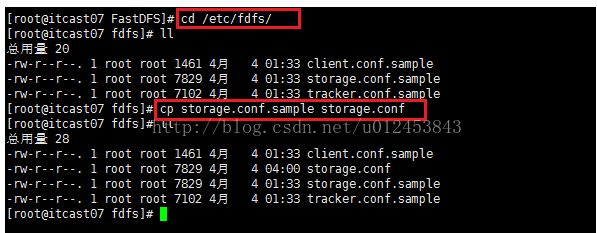
 下面我们便使用命令：/etc/init.d/fdfs\_trackerd start 进行启动，启动之后，我们再查看该目录，发现多了两个目录data和logs，我们可以通过命令：ps -ef | grep fdfs来查看tracker是否正常启动，如下图所示。同理，我们启动一下192.168.156.6上的tracker。



  如果想要停止tracker的话，就使用命令/etc/init.d/fdfs\_trackerd stop。

16.配置storage，按照我们的规划，192.168.156.7、192.168.156.8、192.168.156.9、192.168.156.10四台虚拟机将作为我们的存储节点，四个节点中同一组的配置应该是相同的，也就是192.168.156.7、192.168.156.8一组（组名为group1），192.168.156.9、192.168.156.10一组（组名为group2）。

      首先，进入4台机器的配置文件目录/etc/fdfs，然后进行复制storage.conf.sample并更名为storage.conf，使用的命令：cp storage.conf.sample storage.conf，如下图所示



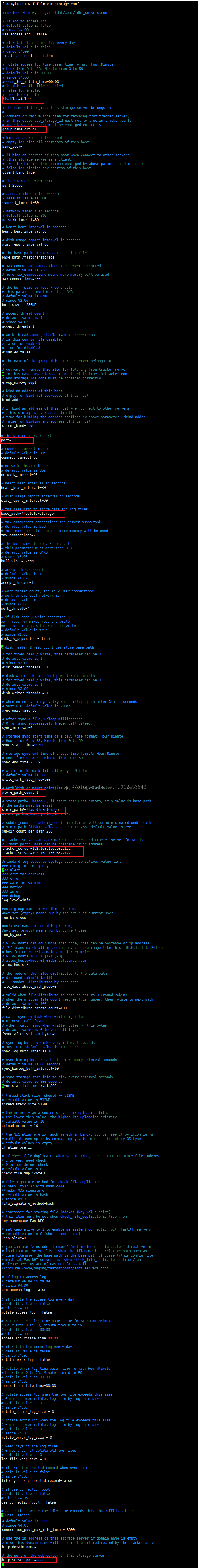
 接下来我们编辑storage.conf文件，对于group1的192.168.156.7、192.168.156.8这两台设备需要修改的配置如下。

       base\_path=/fastdfs/storage

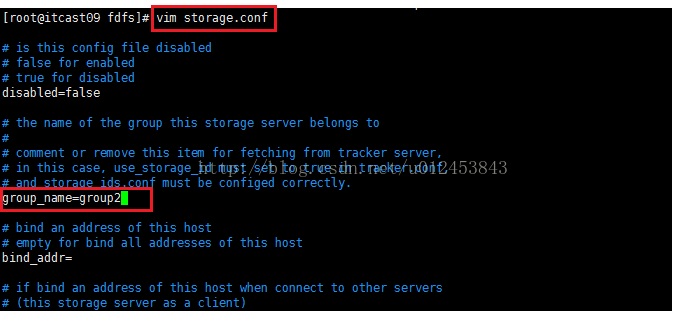
       store\_path0=/fastdfs/storage

       tracker\_server=192.168.156.5:22122

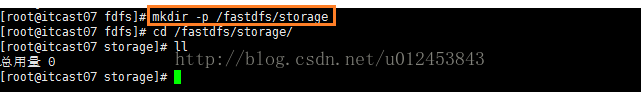
       tracker\_server=192.168.156.6:22122



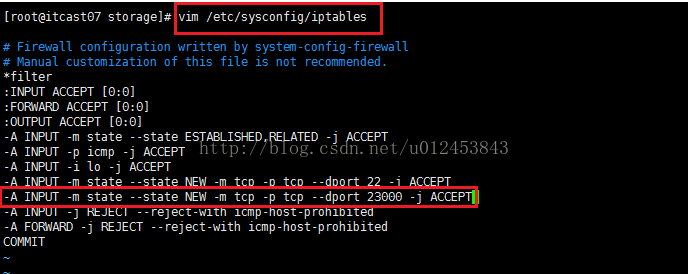
   配置完了192.168.156.7，接下来我们把storage.conf文件复制到192.168.156.8、192.168.156.9、192.168.156.10这三台设备上。其中192.168.156.8这台设备与192.168.156.7同属于group1，因此把配置文件放到它的/etc/fdfs/目录后不用做任何修改。但是192.168.156.9和192.168.156.10这两台设备需要修改一下，修改也非常简单，只需要把group\_name由group1改为group2就可以了，如下图所示。



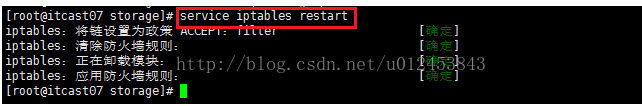
 由于四个配置文件的base\_path=/fastdfs/storage和store\_path0=/fastdfs/storage都配置成了/fastdfs/storage，但是目前我们这四台虚拟机还未创建过该目录，因此我们需要为这四台虚拟机都创建一下该目录,命令：mkdir -p /fastdfs/storage，如下图所示。



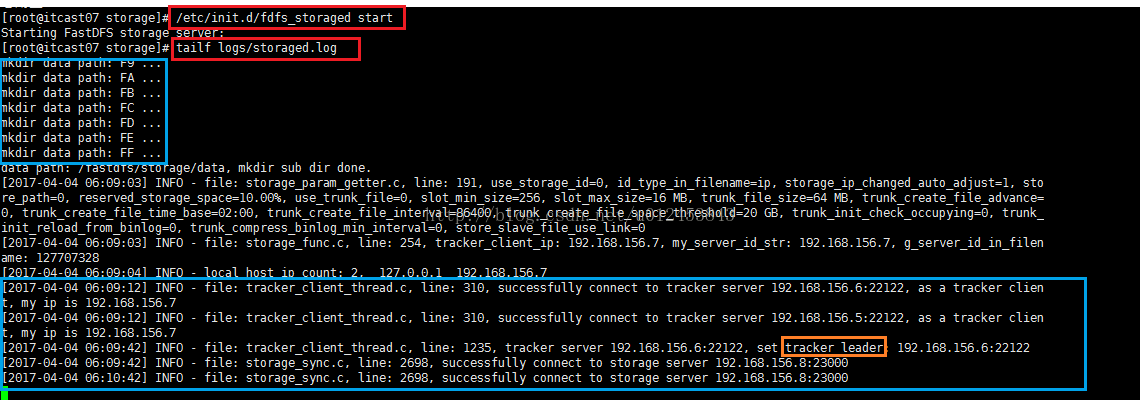
启动storage之前，我们需要先对192.168.156.7、192.168.156.8、192.168.156.9、192.168.156.10这四台虚拟机的防火墙做下配置，添加端口2300，添加语句：-A INPUT -m state --state NEW -m tcp -p tcp --dport 23000 -j ACCEPT，如下图所示。



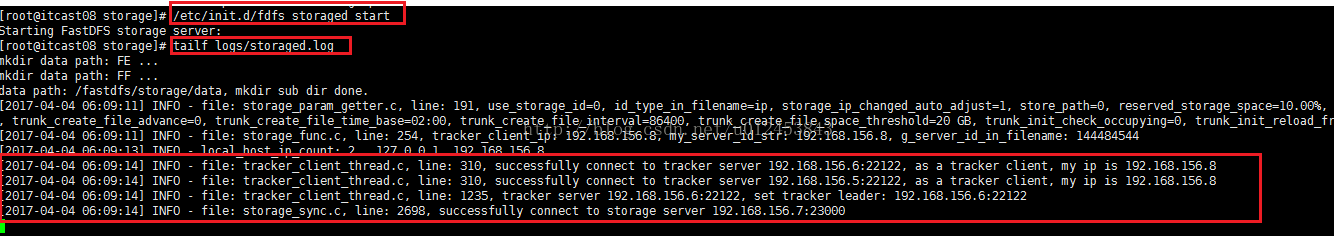
 配置完防火墙之后，我们重启下这四台虚拟机的防火墙，如下图所示。



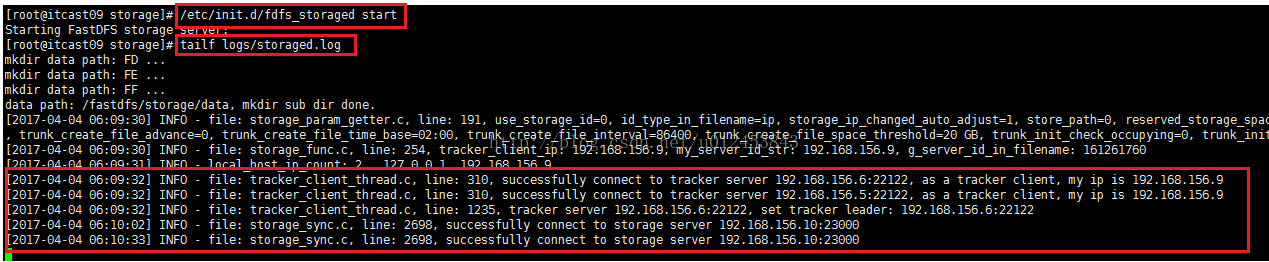
  配置完防火墙之后，我们到storage所在的四台虚拟机的/fastdfs/storage目录下，使用命令：/etc/init.d/fdfs\_storaged start进行启动，然后使用命令：tailf logs/storaged.log来查看启动信息，如下图所示（192.168.156.7这台虚拟机的操作图）。可以看到，192.168.156.7这台设备成功与两个tracker设备连接了，其中选举了192.168.156.6作为tracker集群的leader。192.168.156.7和192.168.156.8这两台虚拟机同属一个分组(group1)，因此可以从下图的信息中看到成功与192.168.156.8连接的信息。



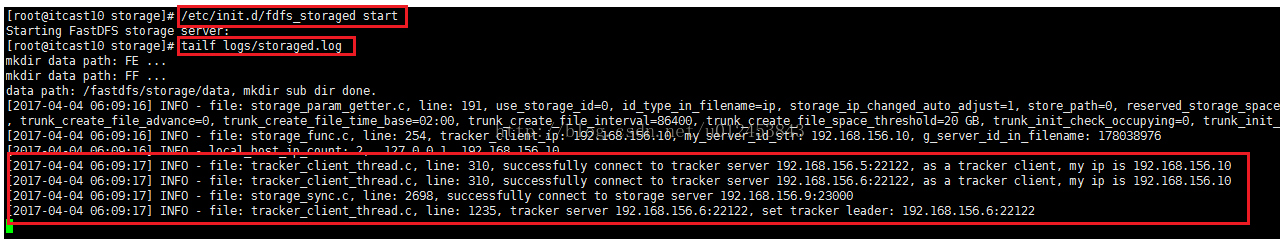
下图是192.168.156.8的启动信息。看到信息几乎一样，只是最后一行显示的是成功与192.168.156.7连接（因为它们同属一组）



下图是192.168.156.9的启动信息。看到的tracker的信息与group1的两台设备一样，只是这台设备与192.168.156.10属于group2，所以它连接的设备是192.168.156.10。



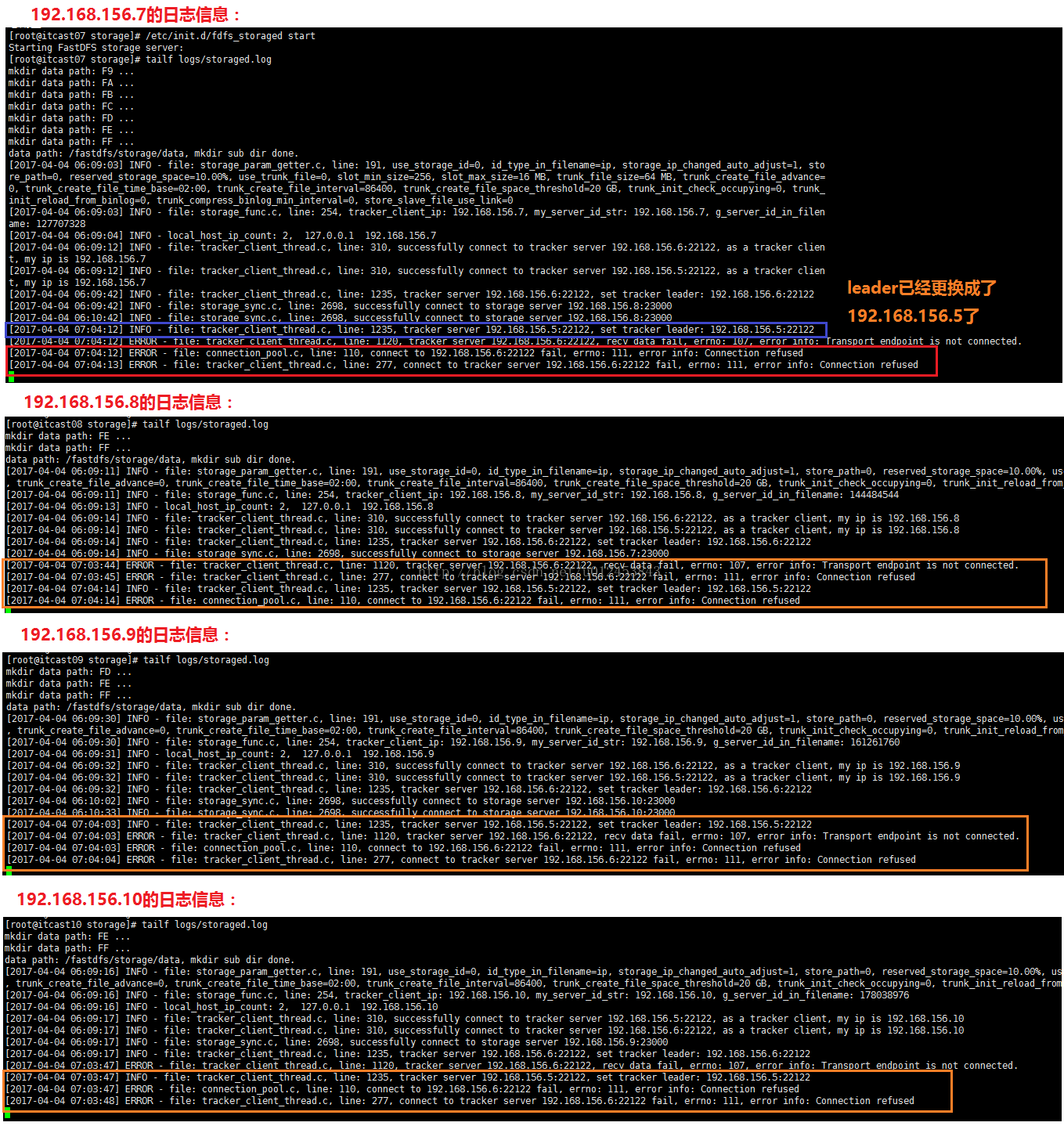
下图是192.168.156.10的启动信息，从下面可以看到成功与tracker还有192.168.156.9连接了。



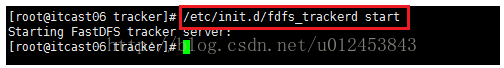
看到上面的信息，说明我们的tracker和storage集群搭建好了，下面我们测试一下tracker的高可用性，从上图我们可以看到192.168.156.6这台设备是作为tracker的leader的，我们现在人为关掉192.168.156.6的tracker。如下图所示。



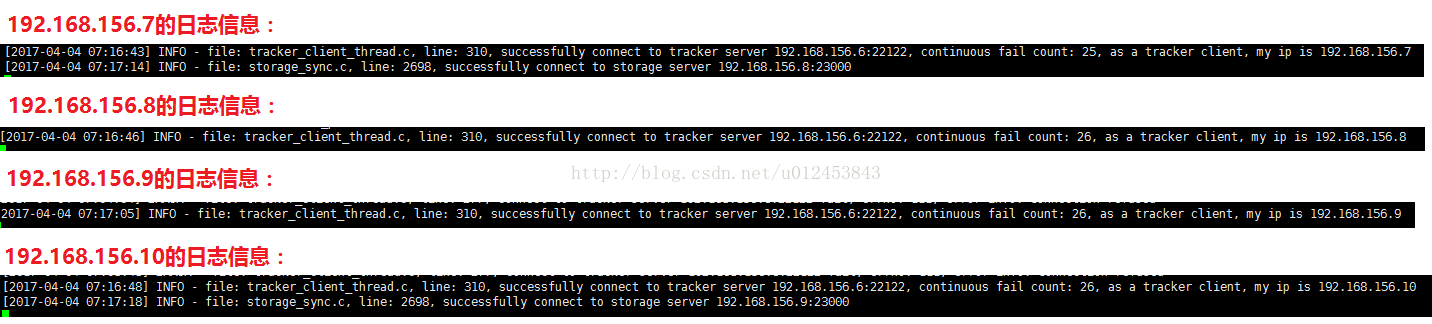
关掉192.168.156.6的tracker之后，我们再看看storage所在的四台虚拟机的日志信息。如下图所示，可以看到192.168.156.5自动切换成leader角色了，而且显示连不上192.168.156.6了（因为已经关了tracker了，所以都连不上了）。



   下面我们再启动192.168.156.6上的tracker，如下图所示。



  启动后，再看看四台设备的日志信息，如下图所示，可以看到，都重新连接上该tracker了。只不过此时tracker的leader依然是192.168.156.5。



      当我们所有的tracker和storage节点都启动成功之后，我们可以在任意的一个存储节点上查看存储集群的信息，命令：/usr/bin/fdfs\_monitor /etc/fdfs/storage.conf，可以看到如下图所示的集群信息。可以看到tracker Server有两个，当前提供服务的是192.168.156.5，group的数量是2，第一组的IP有192.168.156.7和192.168.156.8，第二组的IP有192.168.156.9和192.168.156.10，与我们规划的集群完全一致。

可以使用命令cd /usr/bin/ && ls | grep fdfs来查看fdfs所有的命令，如下图所示。

