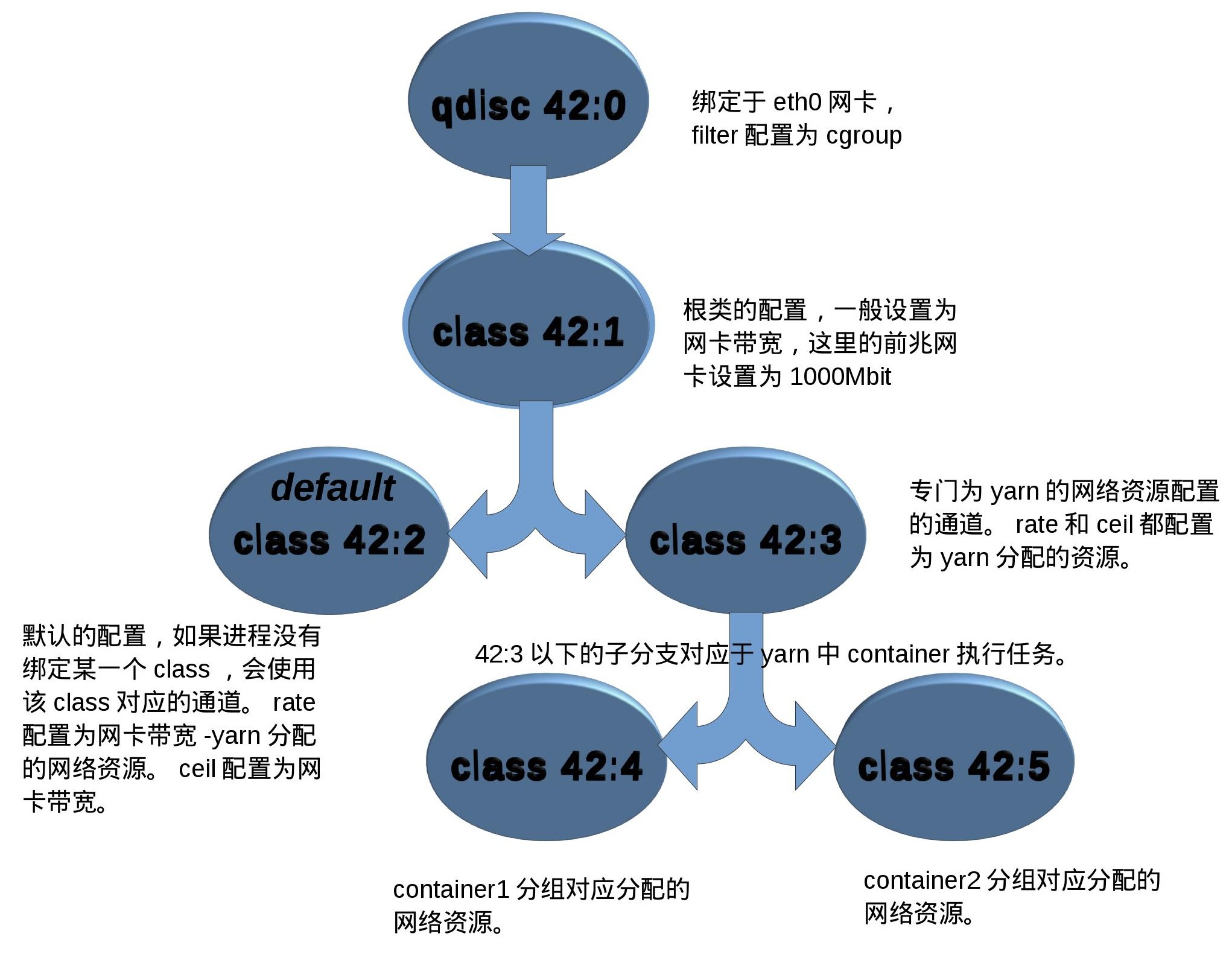
LinuxContainer网络限制实验

一 CGroup关于net\_cls限制的实验

本部分设置由Hadoop关于tc及Cgroup的设置而来。具体的资源配置如下，其中数组表示各个组件的ID号。



1 配置net\_cls环境

1.1 清除资源

tc qdisc del dev eth0 parent root

1.2 总资源设置

tc qdisc add dev eth0 root handle 42: htb default 2

1.3 filter配置

tc filter add dev eth0 parent 42: protocol ip prio 10 handle 1: cgroup

1.4 根class

tc class add dev eth0 parent 42:0 classid 42:1 htb rate 1000Mbit ceil 1000Mbit

1.5 默认class

tc class add dev eth0 parent 42:1 classid 42:2 htb rate 200Mbit ceil 1000Mbit

1.6 yarn的类

tc class add dev eth0 parent 42:1 classid 42:3 htb rate 800Mbit ceil 1000Mbit

2 配置cgroup环境

2.1 创建两个分组

mkdir /sys/fs/cgroup/net\_cls/container1

mkdir /sys/fs/cgroup/net\_cls/container2

2.2 在对应的cgroup分组中写入classid

echo 0x00420004 > /sys/fs/cgroup/net\_cls/container1/net\_cls.classid

echo 0x00420005 > /sys/fs/cgroup/net\_cls/container2/net\_cls.classid

2.3 创建分组的对应的class

tc class add dev eth0 parent 42:3 classid 42:4 htb rate 300Mbit ceil 800Mbit

tc class add dev eth0 parent 42:3 classid 42:5 htb rate 500Mbit ceil 800Mbit

strictMode的参数设置

tc class add dev eth0 parent 42:3 classid 42:4 htb rate 300Mbit ceil 300Mbit

tc class add dev eth0 parent 42:3 classid 42:5 htb rate 500Mbit ceil 500Mbit

2.4 运行具体任务，并将进程号吸入tasks文件。使用scp-182.sh和scp-183.sh进行对比实验。

scp-182.sh的内容:

echo $? > /sys/fs/cgroup/net\_cls/container1/tasks

exec scp a root@172.18.149.182:/root

scp-183.sh的内容:

echo $? > /sys/fs/cgroup/net\_cls/container1/tasks

exec scp a root@172.18.149.182:/root

2.5 清理

tc class del dev eth0 classid 42:4

tc class del dev eth0 classid 42:5

3. 实验的进行

分两组实验进行测量，即strict模式和非strict模式。主要区别在ID为42:3下面的子Class的ceil设置为与rate相同还是设置为yarn的最大值。前者是strict模式，后者是非strict模式。当然实际我们会选用非strict模式。

注: 300Mbit对应的理论最大传输速度为37.5MB/s

500Mbit对应的理论最大传输速度为62.5MB/s

800Mbit对应的理论最大传输速度为100MB/s

1000Mbit对应的理论最大传输速度为125MB/s

但实际都不会达到理论最大值。

3.1 非strcit模式

(1) 首先执行sh scp-182.sh，待稳定后发现速率是92.2MB/s，对应于800Mbit的带宽。

(2) 当同时在另一个终端执行sh scp-183.sh。待稳定后发现之前的速率减低为34.8MB/s，对应于300Mbit的带宽。而新起来的进程速率为57.4MB/s,对应于500Mbit的带宽。



(3)然后关掉第一个进程，第二进程的速率升高，稳定后升到91.6MB/s，对应于800MBit的贷款。

结论: 实际上的进程在空闲的时候可以动态的调整到为yarn设置的最大值。如果进程过多，会根据分配的rate进行网络限制，当然后还有prio进行限制，只是这里暂时没有实验。

3.2 strict模式

(1) 首先执行sh scp-182.sh，待稳定后发现速率是34.8MB/s，对应于300Mbit的带宽。

(2) 当同时在另一个终端执行sh scp-183.sh。发现之前进程速率减低为34.5MB/s，仍然对应于300Mbit的带宽。而新起来的进程速率为57.4MB/s,对应于500Mbit的带宽。



(3)然后关掉第一个进程，第二进程的速率为57.4MB/s，仍然对应于500MBit的带宽。

结论: strict模式下，对网络进行了严格的限制。

3.3 动态调整

使用如下命令，即非strict模式，进开启一个进程，这是发现网络传输速率稳定在92.2MB/s，对应于800Mbit的带宽。

tc class change dev eth0 parent 42:3 classid 42:5 htb rate 500Mbit ceil 800Mbit

在进程运行的期间，执行如下命令更改对应的ceil设置，会发现网络传输速度下降，最终会稳定在57.4MB/s，对应于500Mbit的带宽。值得一提的是，这个过程得到的稳定值非常接近理论值。

tc class change dev eth0 parent 42:3 classid 42:5 htb rate 500Mbit ceil 500Mbit

结论: 可以动态调整网络限制。具体的应用场景为，如果想严格限制网络，根据给定的container的数目进行平分，完全可以动态的调整。

附录1: 一些tc参数的含义

(1) rate rate allocated to this class (class can still borrow) 是一个类保证得到的带宽值,如果有不只一个类,请保证所有子类总和是小于或等于父类

(2) burst max bytes burst which can be accumulated during idle period 突发流量

(3) ceil definite upper class rate (no borrows) 是一个类最大能得到带宽值

(4) prio priority of leaf; lower are served first 是优先权的设置,数值越大,优先权越小,如果是分配剩余带宽,就是数值小的会最优先取得剩余的空闲的带宽权

二 Yarn上Nodemanager的网络限制

具体执行的Yarn应用程序是为AM启动一个容器，容器启动scp.sh的脚本，脚本的内容如下：

(time scp /home/yarn/a 172.18.149.182:/home/yarn/a) &>> /home/yarn/scp.log

a文件的大小是1392404318 (1.3G或1327.9M)

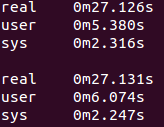
NM中为yarn分配的带宽是800Mbit

(1) 非strict模式

启动一个任务，仅仅一个container享用800Mbit的带宽，理论最大速度是100Mbyte/s,传输一个a文件的理论最快事件为13.279s。实际的传输事件为14.594s，即90.98Mbye/s。说明已经达到了yarn的网络资源限制的效果。实验数据如下:



如果同时启动两个任务，会的到如下的数据。两组scp任务的网速分别为48.95Mbyte/s,48.94Mbyte/s。说明实现了网络资源限制的效果。



(2) strict模式

开启strcit模式，启动一个任务，目前NM设置单个Container为1/50，即最大网络传输速度为2.0Mbyte/s。这里得到的速度为1.84Mbye/s, 说明网络限制生效了。实验数据如下:

