1. ntp使用本地rtc时钟源
2. 编译安装： rpm -qa | grep ntp

没有的话，安装rpm –Uvh ntp\*\*\*.rpm

2,配置：vi /etc/ntp.conf

1. 允许任何客户及都可以同步

将“restrict default nomodify notrap noquery”这行修改成：  
restrict default nomodify notrap

B只允许指定网段的

在restrict default nomodify notrap noquery（表示默认拒绝所有IP的时间同步）之后增加一行：  
restrict 192.168.211.0 mask 255.255.255.0 nomodify notrap

C,不用网络时钟源：

server 127.127.1.0 ///其他的127网段好像有点不行

fudge 127.127.1.0 stratum 8 ////这行可以不需要 stratum表示时钟源层次

3重启ntpd

Service ntpd restates

4,查看自身是否完成了与时钟源的同步

Watch ntpq -p

Every 2.0s: ntpq -p Fri Apr 22 13:49:47 2016

remote refid st t when poll reach delay offset jitter

==============================================================================

\*LOCAL(0) .LOCL. 5 l 44 64 377 0.000 0.000 0.061

5,客户端：

Ntpdate serverip

或者利用crontab来执行  
crontab -e  
0 21 \* \* \* ntpdate 192.168.211.22 >> /root/ntpdate.log 2>&1

1. 检查ntp的版本，如果你使用的是ntp4.2（包括4.2）之后的版本，在restrict的定义中使用了notrust的话，会导致以上错误。  
   使用以下命令检查ntp的版本：  
   # ntpq -c version  
   下面是来自ntp官方网站的说明：  
   The behavior of notrust changed between versions 4.1 and 4.2.  
   In 4.1 (and earlier) notrust meant "Don't trust this host/subnet for time".  
   In 4.2 (and later) notrust means "Ignore all NTP packets that are not cryptographically authenticated." This forces remote time servers to authenticate themselves to your (client) ntpd  
   解决：  
   把notrust去掉。

注意要关掉防火墙 selinux等，最好把客户端的配置文件中的server 也改成指定的server ip

参考<http://blog.csdn.net/skdkjzz/article/details/41678545>

ALTH\_WARN clock skew detected on的字样，说明各个mon节点的时间不同步。解决的办法当然是让各个节点使用同一时间源，确保时间一致，虚拟机的话，可以使用类似vmtools同步宿主机器的时间，也可以配置ntp确保一致。实在没法解决也可以在ceph.conf中新增配置：

mon clock drift allowed = 1 // 1表示1秒钟，这里可以配置小数点。

Crontab用法：

1. cron是进程
2. crontab是文件： /etc/crontab /var/spool/cron/下也存放以用户命名的任务

cron服务每分钟不仅要读一次/var/spool/cron内的所有文件，还需要读一次/etc/crontab

1. crontab也是命令：

crontab -e ：编辑任务，此时会在/var/spool/cron/下生成以用户命名的任务文件

crontab -l :列出任务

crontab –r ： 删除任务

1. 例子：

crontab 文件中每个条目中各个域的意义和格式：

第一列 分钟： 1——59

第二列 小时： 1——23(0表示子夜)

第三列 日 ： 1——31

第四列 月 ： 1——12

第五列 星期： 星期0——6(0表示星期天，1表示星期一、以此类推)

第六列 要运行的命令

我们暂且用C1、C2、C3、C4、C5、C6代表这六列，前面五列通过组合方式来决定执行脚本的频率，最小频率为每分钟执行一次，其中Cn可以用 \* ; \*/n ; T1-T2; a,b,c; 四种形式来表示：

当 C1 为 \* 时表示每分钟都要执行脚本，C2 为 \* 时表示每小时都要执行程式，依次类推.....

当 C1 为 T1-T2 时表示从第 T1 分钟到第 T2 分钟这段时间内要执行，C2 为 T1-T2 时表示从第 T1 到第 T2 小时都要执行，依次类推....

当 C1 为 \*/n 时表示每 n 分钟的时间间隔执行一次，C2 为 \*/n 表示每隔n小时的间隔执行一次，依次类推.....

当 C1 为 a, b, c,... 时表示第 a, b, c,... 分钟要执行，C2 为 a, b, c,... 时表示第 a, b, c...个小时要执行，依次类推....

下面列举几个例子供大家参考

1： 59 23 \* \* \* /home/oracle/scripts/alert\_log\_archive.sh >/dev/null 2>&1

表示每天23点59分执行脚本/home/oracle/scripts/alert\_log\_archive.sh

2: \*/5 \* \* \* \* /home/oracle/scripts/monitoring\_alert\_log.sh >/dev/null 2>&1

表示每5分钟执行一次脚本/home/oracle/scripts/monitoring\_alert\_log.sh

3： 0 20 \* \* 1-5 mail -s "\*\*\*\*\*\*\*\*\*\*" kerry@domain.name < /tmp/maildata

周一到周五每天下午 20:00 寄一封信给 kerry@domain.name

..............................................

关于 >/dev/null 2>&1 的解释：

0表示键盘输入

1表示标准输出

2表示错误输出.

我们首先创建test.sh脚本如下：

#! /bin/sh

echo "hello, everybody, now is " `date`

date >> test.txt

然后添加作业

\* \* \* \* \* /home/oracle/test.sh >/home/oracle/log.txt & 默认值为1，即和下面命令一致

\* \* \* \* \* /home/oracle/test.sh 1>/home/oracle/log.txt &

\* \* \* \* \* /home/oracle/test.sh 2>/home/oracle/log.txt &

\* \* \* \* \* /home/oracle/test.sh 2>/home/oracle/log.txt 2>&1 &

1,2将tesh.sh 命令输出重定向到log.txt, 即输出内容不打印到屏幕上，而是输出到log.txt文件中。如果你需要追加而不是覆盖，可以用 >>代替>

2>&1 是将错误输出重定向到标准输出。 然后将标准输入重定向到文件log.txt。

&1 表示的是文件描述1，表示标准输出，如果这里少了&就成了数字1，就表示重定向到文件1。

注意事项：

配置定时任务时，需要注意两个问题:

1: 在SHELL中设置了必要的环境变量；例如一个shell脚本手工执行OK，但是配置成后台作业执行时，获取不到ORACLE的环境变量，这是因为crontab环境变量问题，Crontab的环境默认情况下并不包含系统中当前用户的环境。所以，你需要在shell脚本中添加必要的环境变量的设置

2: 尽量所有的文件都采用完全路径方式，避免使用相对路径。

5，/etc/crontab中的文件格式：

cron服务每分钟不仅要读一次/var/spool/cron内的所有文件，还需要读一次/etc/crontab,因此我们配置这个文件也能运用cron服务做一些事情。用crontab配置是针对某个用户的，而编辑/etc/crontab是针对系统的任务。此文件的文件格式是：

　　SHELL=/bin/bash  
　　PATH=/sbin:/bin:/usr/sbin:/usr/bin  
　　MAILTO=root //如果出现错误，或者有数据输出，数据作为邮件发给这个帐号  
　　HOME=/  
　　# run-parts  
　　01 \* \* \* \* root run-parts /etc/cron.hourly //每个小时去执行一遍/etc/cron.hourly内的脚本  
　　02 4 \* \* \* root run-parts /etc/cron.daily //每天去执行一遍/etc/cron.daily内的脚本  
　　22 4 \* \* 0 root run-parts /etc/cron.weekly //每星期去执行一遍/etc/cron.weekly内的脚本  
　　42 4 1 \* \* root run-parts /etc/cron.monthly //每个月去执行一遍/etc/cron.monthly内的脚本

　　使用者 运行的路径

　　注意"run-parts"这个参数了，如果去掉这个参数的话，后面就可以写要运行的某个脚本名，而不是文件夹名了。

如： \*/10 \* \* \* \* root /bin/sh /etc/cron.daily/keep\_rsyslog.sh