**Heartbeat原理与实战**

# Heartbeat的组成与原理

Linux-HA的全称是High-Availability Linux，它是一个开源项目。这个开源项目的目标是：通过社区开发者的共同努力，提供一个增强Linux可靠性（reliability）、可用性（availability）和可服务性（serviceability）（RAS）的群集解决方案。其中Heartbeat就是Linux-HA项目中的一个组件，也是目前开源HA项目中最成功的一个例子，它提供了所有HA软件所需要的基本功能，比如心跳检测和资源接管、监测群集中的系统服务、在群集中的节点间转移共享IP地址的所有者等。

## Heartbeat的组成

Heartbeat提供了高可用集群最基本的功能，例如，节点间的内部通信方式、集群合作管理机制、监控工具和失效切换功能等。目前的最新版本是Heartbeat 2.x，这里的讲述也是以Heartbeat 2.x为主。下面介绍Heartbeat 2.0的内部组成，主要分为以下几大部分。

heartbeat：节点间通信检测模块；

ha-logd：集群事件日志服务；

CCM（Consensus Cluster Membership）：集群成员一致性管理模块；

LRM（Local Resource Manager）：本地资源管理模块；

Stonith Daemon：使出现问题的节点从集群环境中脱离；

CRM（Cluster Resource Management）：集群资源管理模块；

Cluster policy engine：集群策略引擎；

Cluster transition engine：集群转移引擎；

Heartbeat仅仅是个HA软件，它仅能完成心跳监控和资源接管，不会监视它控制的资源或应用程序。要监控资源和应用程序是否运行正常，必须使用第三方的插件，例如ipfail、Mon和Ldirector等。Heartbeat自身包含了几个插件，分别是ipfail、Stonith和Ldirectord，介绍如下。

ipfail的功能直接包含在Heartbeat里面，主要用于检测网络故障，并做出合理的反应。为了实现这个功能，ipfail使用ping节点或者ping节点组来检测网络连接是否出现故障，从而及时做出转移措施。

Stonith插件可以在一个没有响应的节点恢复后，合理接管集群服务资源，防止数据冲突。当一个节点失效后，会从集群中删除。如果不使用Stonith插件，那么失效的节点可能会导致集群服务在多于一个节点运行，从而造成数据冲突甚至是系统崩溃。因此，使用Stonith插件可以保证共享存储环境中的数据完整性。

Ldirector是一个监控集群服务节点运行状态的插件。Ldirector如果监控到集群节点中某个服务出现故障，就屏蔽此节点的对外连接功能，同时将后续请求转移到正常的节点提供服务，这个插件经常用在LVS负载均衡集群中。

同样，对于操作系统自身出现的问题，Heartbeat也无法监控。如果主节点操作系统挂起，一方面可能导致服务中断，另一方面由于主节点资源无法释放，而备份节点却接管了主节点的资源，此时就发生了两个节点同时争用一个资源的状况。

针对这个问题，就需要在Linux内核中启用一个叫watchdog的模块。watchdog是一个Linux内核模块，它通过定时向/dev/watchdog设备文件执行写操作，从而确定系统是否正常运行。如果watchdog认为内核挂起，就会重新启动系统，进而释放节点资源。

在Linux中完成watchdog功能的软件叫softdog。softdog维护一个内部计时器，此计时器在一个进程写入/dev/watchdog设备文件时更新。如果softdog没有看到进程写入/dev/watchdog文件，就认为内核可能出了故障。watchdog超时周期默认是一分钟，可以通过将watchdog集成到Heartbeat中，从而通过Heartbeat来监控系统是否正常运行。

## Heartbeat的原理

**集群成员一致性管理模块（CCM）**

CCM用于管理集群节点成员，同时管理成员之间的关系和节点间资源的分配。Heartbeat模块负责检测主次节点的运行状态，以决定节点是否失效。ha-logd模块用于记录集群中所有模块和服务的运行信息。

**本地资源管理器（LRM）**

LRM负责本地资源的启动、停止和监控，一般由LRM守护进程lrmd和节点监控进程Stonith Daemon组成。lrmd守护进程负责节点间的通信；Stonith Daemon通常是一个Fence设备，主要用于监控节点状态，当一个节点出现问题时处于正常状态的节点会通过Fence设备将其重启或关机以释放IP、磁盘等资源，始终保持资源被一个节点拥有，防止资源争用的发生。

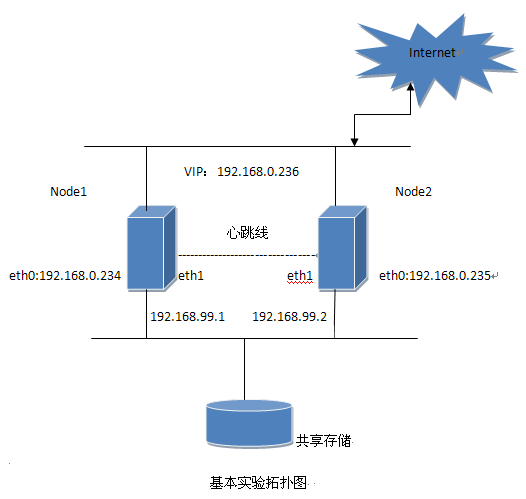
**集群资源管理模块（CRM）**

CRM用于处理节点和资源之间的依赖关系，同时，管理节点对资源的使用，一般由CRM守护进程crmd、集群策略引擎和集群转移引擎3个部分组成。集群策略引擎（Cluster policy engine）具体实施这些管理和依赖；集群转移引擎（Cluster transition engine）监控CRM模块的状态，当一个节点出现故障时，负责协调另一个节点上的进程进行合理的资源接管。

在Heartbeat集群中，最核心的是Heartbeat模块的心跳监测部分和集群资源管理模块的资源接管部分。心跳监测一般由串行接口通过串口线来实现，当然，也可以用交叉网线连接。两个节点之间通过串口线相互发送报文来告诉对方自己当前的状态。如果在指定的时间内未受到对方发送的报文，就认为对方失效，这时资源接管模块将启动，用来接管运行在对方主机上的资源或者服务。

# 安装Heartbeat前的准备

## 实验环境规划



## 创建虚拟共享磁盘

1. 创建存放共享磁盘文件的目录，若虚拟机软件安装路径下的磁盘够大也可以不用创建此目录，直接将要创建的磁盘文件放在相应目录即可；

2. 创建共享磁盘文件

在虚拟机软件的安装目录下，有个vmware-vdiskmanager.exe文件（老版本用plainmaker.exe），  
    把它复制到共享磁盘柜目录下，创建共享磁盘（老版本共享磁盘文件的扩展名为.pln）， 注意，此种方法有可能不成功，若不成功，可进入到虚拟机软件的安装目录，在其下创建即可：

D:\vmware系统\共享磁盘> **vmware-vdiskmanager.exe -c -s 200Mb -a lsilogic -t 2 quorum.vmdk #仲裁盘**  
     D:\vmware系统\共享磁盘> **vmware-vdiskmanager.exe -c -s 2Gb -a lsilogic -t 2 sharedisk.vmdk #数据盘**

1. 确认是否有磁盘文件生成：

创建完毕后，目录下有四个新文件：  
            **quorum-flat.vmdk**  
**quorum.vmdk**  
**sharedisk-flat.vmdk**  
**sharedisk.vmdk**

1. 确认成功生成后，用虚拟机软件在相应系统中，在虚拟系统关闭的状态下，添加刚刚生成的磁盘，并在高级选项中分别选择不同的通道，我们这里选择给仲裁盘选1:5，给数据盘选1:6，两个系统都要添加，保存配置；
2. 进入到相应虚拟机系统目录下（本实验中的node1及node2）找到.vmx后辍的文件，用记事本打开添加下面两行内容：

disk.locking=”FALSE”

disklib.dataCacheMaxSize=”0”

1. 先打开node1系统，格式化磁盘并创建文件系统，mkdir /sharedisk ，创建挂载目录为后续工作做准备；
2. 关闭node1,打开node2系统，mkdir /sharedisk ，查看相应的磁盘设备号后可直接挂载使用，此时可开启node1系统;

注意：

Node1与node2节点系统硬件配置应尽量保持一致，至少应该保证磁盘一致性，比如，配置heartbeat之前都有一块或几块磁盘，我们这里之前都只有一块磁盘。

手动挂载仲裁盘，此盘不可以人为写入数据，可以将此盘写到/etc/fstab下，让系统自动挂载。

共享数据盘会在heartbeat 的haresources文件中自动挂载，因此不需要手工挂载。

## 配置Apache

为更加直观的观测高可用的测试结果，我们这里配置基于apache的web应用。

简述下配置过程：

1. 安装软件包，为方便测试，这里直接安装rpm包：

[root@node1 ~]#yum install httpd

1. 编辑主配置文件httpd.conf

[root@node1 ~]#vim /etc/httpd/conf/httpd.conf

ServerName 192.168.0.234:80

DocumentRoot "/sharedisk/www" #将web页目录更改到共享磁盘路径下；

<Directory "/sharedisk/www">

1. 手动挂载共享磁盘，创建web目录及测试页：

[root@node1 ~]#mount /dev/sdc1 /sharedisk

[root@node1 ~]#mkdir /sharedisk/www

[root@node1 ~]#vim /sharedisk/www/index.html

[root@node1 ~]#cat /sharedisk/www/index.html

welcome to my web station.

1. 启动apache

[root@node1 ~]#service httpd start

[root@node1 ~]#lsof -n -i:80

## 创建用户和用户组

同多数其它应用服务一样，heartbeat也需要用户和用户组的支持，不过，这里需要我们手动创建：

[root@node1 ~]# groupadd -g 694 haclient

[root@node1 ~]# useradd -u 694 -g haclient hacluster

注意，同样需要在node2上执行相同操作。

## 编辑Hosts文件

[root@node1 ~]#vi /etc/hosts #加入两个节点的IP及对应的主机名如下：

192.168.0.234 node1

192.168.0.235 node2

# 安装Heartbeat

## 获取Heartbeat

Heartbeat到目前为止发行了两个主版本，即Heartbeat 1.x和Heartbeat 2.x 。Heartbeat 1.x仅仅允许创建两个节点的集群，提供基本的高可用性failover服务。Heartbeat 2.x提供了增强的特性，允许创建多个节点的集群，且支持模块化结构的配置方法——集群资源管理器（Cluster Rescource Manager-CRM），CRM可以支持最多16个节点，我们这里选用的版本为Heartbeat-2.0.6。

Heartbeat软件包可以通过在网络上搜索下载，也可以从以下站点下载：<http://liunx-ha.org/download/index.html> 。

安装Heatbeat前必须先安装libnet包，可以从下面的站点下载: <http://www.packetfactory.net/projects/libnet/> 。

libnet提供了一些高层的api，让应用程序开发者可以修改网络包，我们这里下载的版本为：libnet-1.1.2.1 。

## 安装Heartbeat

接下来分别对libnet和heartbeat进行安装，安装过程很简单，只需解压编译，安装即可，在两个节点执行相同的操作，下面是在node1上的安装过程：

1. 安装libnet

[root@node1 software]# pwd

/tmp/software

[root@node1 software]# ll

总计 4200

-rw-r--r-- 1 root root 3267773 08-25 18:51 heartbeat-2.0.6.tar.gz

-rw-r--r-- 1 root root 1021236 08-25 18:51 libnet-1.1.2.1.tar.gz

[root@node1 software]# tar xf libnet-1.1.2.1.tar.gz

[root@node1 sofeware]# cd libnet

[root@node1 sofeware]#./configure

[root@node1 sofeware]#make

[root@node1 sofeware]#make install

1. 安装heartbeat

[root@node1 software]# tar xf heartbeat-2.0.6.tar.gz

[root@node1 sofeware]# cd heartbeat-2.0.6

[root@node1 heartbeat-2.0.6]#./ConfigureMe configure \

–disable-swig –disable-snmp-subagent

[root@node1 heartbeat-2.0.6]#make

[root@node1 heartbeat-2.0.6]#make install

[root@node1 heartbeat-2.0.6]# cp doc/ha.cf doc/authkeys doc/haresources /etc/ha.d/ #Heartbeat的主要配置文件有ha.cf,authkeys和haresources，在Heartbeat安装后，默认并没有这3个文件，可以从官网上下载，也可以从解压出来的源码目录中找到，所以我们这里直接在源码目录中拷贝即可。

# 配置Heartbeat

Heartbeat的主要配置文件有ha.cf,authkeys和haresources，下面将分别详细介绍：

## 配置主配置Ha.cf

以下将对主配置文件ha.cf的基本常用选项作详细说明：

[root@node1 ~]# vim /etc/ha.d/ha.cf

#logfacility local0 #可注释掉此选项，开启下面的日志路径；

logfile /var/log/ha-log #设置heartbeat日志存放位置；

keepalive 2 #设定心跳(监测)时间时间为2秒;

warntime 5 #连续多长时间联系不上后开始警告提示;

deadtime 20 #连续多长时间联系不上后认为对方挂掉了（单位是妙）;

initdead 120 #这里主要是给重启后预留的一段忽略时间段（比如：重启后启动网络等，如果在网络还没有通，keepalive检测肯定通不过，但这时候并不能切换）,此值至少为deadtime的两倍；

udpport 694 #设置广播通信的端口，默认为694；

baud 19200 #设置串行通讯的波特率；

bcast eth1 #指明心跳使用以太网的广播方式，并且在eth1口上进行广播；

auto\_failback off #恢复正常后是否需要再自动切换回来，此处off说明恢复后不需要切换；

node node1 #主节点主机名，可以通过“uname -n”查看；

node node2 #备用节点主机名；

ping 192.168.0.254 #测试网络连通性，此处自定义，一般设为网关地址，但要保证是通的；

respawn hacluster /usr/lib/heartbeat/ipfail #可选，列出和heartbeat一起启动和关闭的进程；

## 配置资源文件Haresources

Haresources文件用于指定双机系统的主节点、集群IP、子网掩码、广播地址以及启动的服务等集群资源。

下面针对本测试项目作如下编辑：

[root@node1 ~]# vim /etc/ha.d/haresources

node1 IPaddr::192.168.0.236/24/eth0/ Filesystem::/dev/sdc1::/sharedisk::ext3 httpd

##上面的192.168.0.236为虚拟出的VIP，自动挂载磁盘的同时启动httpd服务；

这里有几个注意事项：

1) 资源组的第一列是我们在ha.cf配置文件中的node之一，而且应该是当前准备作为primary节点的那一个node；

2）每一行代表一个资源组，如果一行写不下可以用" "换行；

3）资源组启动顺序是从左往右，关闭的顺序是从右往左；

4）脚本的参数通过::来分隔和传递；

5）一个资源组里面不同资源之间以空格分隔；

6）不同的资源组之间没有必然关系；

7）每个资源都是一个角本，可以是在/etc/init.d目录下面的，也可以是/usr/local/etc/ha.d/resource.d目录下面的角本。这些角本必须要支持xxx start；xxx stop;模式；

8）关于service IP的资源设置格式详见haresources文件；

9）如果mysql是编译安装的话， 则需要修改/etc/init.d/mysql文件中的basedir和datadir两个参数。

## 配置双机互联认证文件Authkeys

Authkeys文件用于设定Heartbeat的认证方式，共有3种：crc,md5和sha1。三种认证方式的安全性依次提高，占用系统资源也依次增加，如果集群运行于较为安全的网络上，可以选择使用crc方式，如果HA的每个节点的硬件配置都很高，建议使用sha1，这种认证方式的安全级别最高，md5认证方式介于两者之间，我们这里选用crc的认证方式，设置如下：

[root@node1 ~]# vim /etc/ha.d/authkeys

auth 1

1 crc

注意，需要对此文件设置文件权限，如下：

[root@node1 ~]#chmod 600 /etc/ha.d/authkeys

## 配置备份节点的Heartbeat

在备份节点上也需要安装Heartbeat，安装方式与在主节点安装过程一样，这里不再重述，依次安装libnet和heartbeat源码包，安装完毕，在备份节点上使用scp命令把主节点配置文件传输到备份节点即可，注意，配置文件不需作任何更改：

[root@node2 ~]# scp -r node1:/etc/ha.d/\* /etc/ha.d/

## 设置主节点和备份节点时间同步

因为节点之间的监控是通过设定时间来实现的，所以就确保主备节点的时间一致，主备节点之间的系统时间相差在10秒以内是正常的，但相差再大些就可能HA环境故障。解决办法就是设置时间同步，我们这里采用在主节点上搭建NTP服务器，让从节点同步主节点的时间。

这里简述下配置关键点，若要详细了解NTP，可参考其它相关说明文档。

分别在主备节点上安装ntp的rpm包，这在系统光盘里可以找到：

[root@node1 ~]#rpm -ivh /media/Server/ntp-4.2.2p1-9.el5\_4.1.i386.rpm

主节点node1上的配置：

[root@node1 ~]#vim /etc/ntp.conf

restrict 192.168.0.0  mask 255.255.255.0   nomodify  notrap

[root@node1 ~]#/etc/init.d/ntpd start

从节点node2上的配置：

[root@node2 ~]# vim /etc/ntp.conf

server 192.168.0.234  prefer   iburst  minpoll 6 maxpoll 8

[root@node2 ~]#/etc/init.d/ntpd start

可用命令：watch  ntpq -p 查看时间同步情况

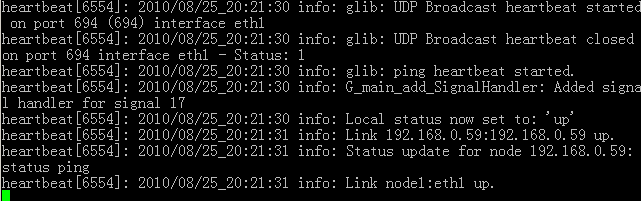
# 启动与测试

1. 开启node1主节点的heartbeat

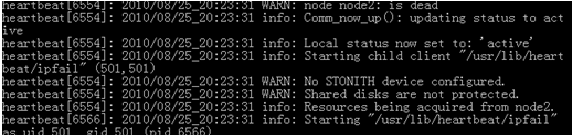
开启日志，实时查看heartbeat的状态，同时启动heartbeat;

[root@node1 ~]#tail -f /var/log/ha-log

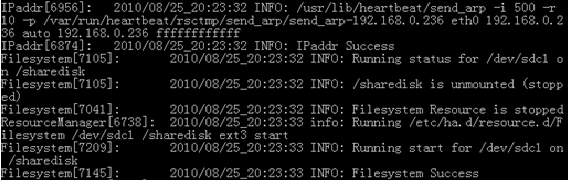
[root@node1 ~]# /etc/init.d/heartbeat start



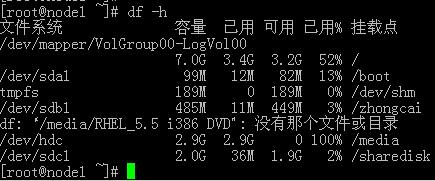
可以看到heartbeat正在进行初始化配置，日志信息到这里会暂停一段时间，等待120秒之后会继续输出日志，如下：



由于node2节点没有开启heartbeat，所以系统给出警告提示“node2: is dead”,接下来会挂载共享磁盘：



提示挂载成功，可以作以下验证：



可以看到共享磁盘/dev/sdc1已成功被挂载到指定的/sharddisk下；

1. 开启node2备用节点的heartbeat

[root@node2 ~]# /etc/init.d/heartbeat start

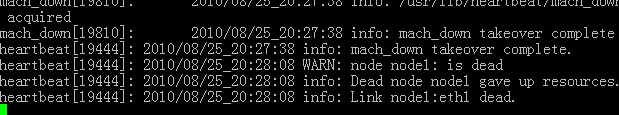
查看磁盘挂载情况：



由于共享磁盘已被主节点挂载使用，此时备用节点并没有挂载，它会在node1节点宕机或掉线时，将heartbeat接管过来的同时再去挂载共享磁盘；

下面关掉主节点的heartbeat：

[root@node1 ~]# /etc/init.d/heartbeat stop



可以看到node1的heartbeat已经关闭，下面再次查看磁盘的挂载情况：



可以看到node2已成功将共享磁盘挂载到/sharedisk下；

1. 访问测试

可以通过多次切换heartbeat服务，测试web应用：



通过多次服务切换证实，web应用可以持续提供服务，实现了高可用。