

## 第一篇：概念篇

### 一、什么是 RHCS

RHCS 是 Red Hat Cluster Suite 的缩写，也就是红帽子集群套件，RHCS 是一个能够提供高可用性、高可靠性、负载均衡、存储共享且经济廉价的集群工具集合，它将集群系统中三大集群架构融合一体，可以给 web 应用、数据库应用等提供安全、稳定的运行环境。

更确切的说，RHCS 是一个功能完备的集群应用解决方案，它从应用的前端访问到后端的数据存储都提供了一个行之有效的集群架构实现，通过 RHCS 提供的这种解决方案，不但能保证前端应用持久、稳定的提供服务，同时也保证了后端数据存储的安全。

RHCS 提供了集群系统中三种集群构架，分别是高可用性集群、负载均衡集群、存储集群。

### 二、RHCS 提供的三个核心功能

高可用集群是 RHCS 的核心功能。当应用程序出现故障，或者系统硬件、网络出现故障时，应用可以通过 RHCS 提供的高可用性服务管理组件自动、快速从一个节点切换到另一个节点，节点故障转移功能对客户端来说是透明的，从而保证应用持续、不间断的对外提供服务，这就是 RHCS 高可用集群实现的功能。

RHCS 通过 LVS (Linux Virtual Server) 来提供负载均衡集群，而 LVS 是一个开源的、功能强大的基于 IP 的负载均衡技术，LVS 由负载调度器和服务访问节点组成，通过 LVS 的负载调度功能，可以将客户端请求平均的分配到各个服务节点，同时，还可

以定义多种负载分配策略，当一个请求进来时，集群系统根据调度算法来判断应该将请求分配到哪个服务节点，然后，由分配到的节点响应客户端请求，同时，LVS 还提供了服务节点故障转移功能，也就是当某个服务节点不能提供服务时，LVS 会自动屏蔽这个故障节点，接着将失败节点从集群中剔除，同时将新来此节点的请求平滑的转移到其它正常节点上来；而当此故障节点恢复正常后，LVS 又会自动将此节点加入到集群中去。而这一系列切换动作，对用户来说，都是透明的，通过故障转移功能，保证了服务的不间断、稳定运行。

RHCS 通过 GFS 文件系统来提供存储集群功能，GFS 是 Global File System 的缩写，它允许多个服务同时去读写一个单一的共享文件系统，存储集群通过将共享数据放到一个共享文件系统中从而消除了在应用程序间同步数据的麻烦，GFS 是一个分布式文件系统，它通过锁管理机制，来协调和管理多个服务节点对同一个文件系统的读写操作。

### 三、RHCS 集群的组成

RHCS 是一个集群工具的集合，主要有下面几大部分组成：

#### **集群构架管理器**

这是 RHCS 集群的一个基础套件，提供一个集群的基本功能，使各个节点组成集群在一起工作，具体包含分布式集群管理器（CMAN）、成员关系管理、锁管理（DLM）、配置文件管理（CCS）、栅设备（FENCE）。

#### **高可用服务管理器**

提供节点服务监控和服务故障转移功能，当一个节点服务出现故障时，将服务转移到另一个健康节点。

#### **集群配置管理工具**

RHCS 最新版本通过 LUCI 来配置和管理 RHCS 集群, LUCI 是一个基于 web 的集群配置方式, 通过 luci 可以轻松的搭建一个功能强大的集群系统。

### **Linux Virtual Server**

LVS 是一个开源的负载均衡软件, 利用 LVS 可以将客户端的请求根据指定的负载策略和算法合理的分配到各个服务节点, 实现动态、智能的负载分担。

RHCS 除了上面的几个核心构成, 还可以通过下面一些组件来补充 RHCS 集群功能。

### **Red Hat GFS (Global File System)**

GFS 是 Redhat 公司开发的一款集群文件系统, 目前的最新版本是 GFS2, GFS 文件系统允许多个服务同时读写一个磁盘分区, 通过 GFS 可以实现数据的集中管理, 免去了数据同步和拷贝的麻烦, 但 GFS 并不能孤立的存在, 安装 GFS 需要 RHCS 的底层组件支持。

### **Cluster Logical Volume Manager**

Cluster 逻辑卷管理, 即 CLVM, 是 LVM 的扩展, 这种扩展允许 cluster 中的机器使用 LVM 来管理共享存储。

### **iSCSI**

iSCSI 是一种在 Internet 协议上, 特别是以太网上进行数据块传输的标准, 它是一种基于 IP Storage 理论的新型存储技术, RHCS 可以通过 iSCSI 技术来导出和分配共享存储的使用。

### **Global Network Block Device**

全局网络模块, 简称 GNBD, 是 GFS 的一个补充组件, 用于 RHCS 分配和管理共享存储, GNBD 分为客户端和服务端, 在服务端 GNBD 允许导出多个块设备或者 GNBD 文件, 而 GNBD 客户端通过导入这些导出的块设备或者文件, 就可以把它们当作本地块设备使用。由于现在 GNBD 已经停止了开发, 所以使用 GNBD 的越来越少。

#### 四、RHCS 集群结构

RHCS 集群从整体上分为三大部分，负载均衡集群、高可用性集群、存储集群，如图 1 所示：

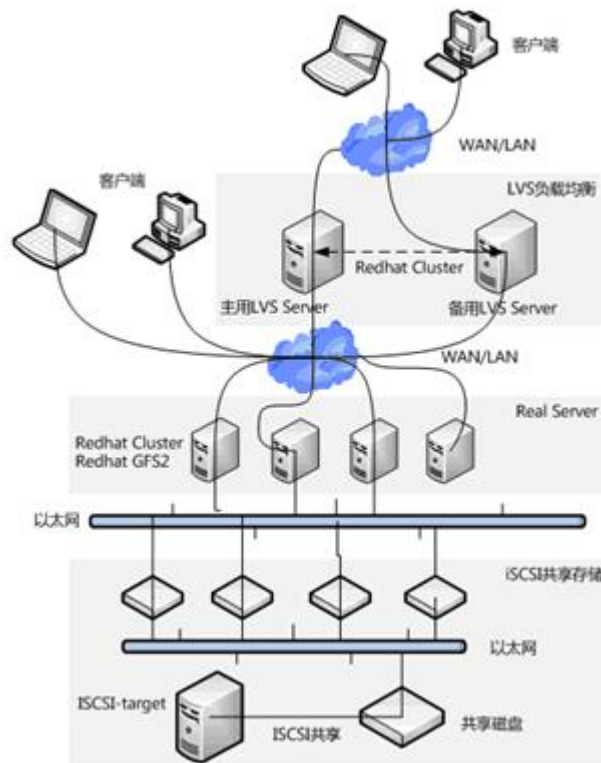


图 1

图 1 是典型的 RHCS 集群拓扑结构：整个拓扑结构分为三个层面：

最上层是 LVS 负载均衡层，中间一层是 Real Server 层，就是服务节点部分，最后一层是共享存储层，主要用于给 GFS 文件系统提供共享存储空间。

#### 五、RHCS 集群运行原理及功能介绍

### 1、 分布式集群管理器（CMAN）

Cluster Manager，简称 CMAN，是一个分布式集群管理工具，它运行在集群的各个节点上，为 RHCS 提供集群管理任务。

CMAN 用于管理集群成员、消息和通知。它通过监控每个节点的运行状态来了解节点成员之间的关系，当集群中某个节点出现故障，节点成员关系将发生改变，CMAN 及时将这种改变通知底层，进而做出相应的调整。

### 2、 锁管理（DLM）

Distributed Lock Manager，简称 DLM，表示一个分布式锁管理器，它是 RHCS 的一个底层基础构件，同时也为集群提供了一个公用的锁运行机制，在 RHCS 集群系统中，DLM 运行在集群的每个节点上，GFS 通过锁管理器的锁机制来同步访问文件系统元数据。CLVM 通过锁管理器来同步更新数据到 LVM 卷和卷组。

DLM 不需要设定锁管理服务器，它采用对等的锁管理方式，大大的提高了处理性能。同时，DLM 避免了当单个节点失败需要整体恢复的性能瓶颈，另外，DLM 的请求都是本地的，不需要网络请求，因而请求会立即生效。最后，DLM 通过分层机制，可以实现多个锁空间的并行锁模式。

### 3、 配置文件管理（CCS）

Cluster Configuration System，简称 CCS，主要用于集群配置文件管理和配置文件在节点之间的同步。CCS 运行在集群的每个节点上，监控每个集群节点上的单一配置文件 `/etc/cluster/cluster.conf` 的状态，当这个文件发生任何变化时，都将此变化更新到集群中的每个节点，时刻保持每个节点的配置文件同步。例如，管理员在节点 A 上更新了集群配

置文件，CCS 发现 A 节点的配置文件发生变化后，马上将此变化传播到其它节点上去。

rhcs 的配置文件是 `cluster.conf`，它是一个 xml 文件，具体包含集群名称、集群节点信息、集群资源和服务信息、fence 设备等，这个会在后面讲述。

#### 4、栅设备（FENCE）

FENCE 设备是 RHCS 集群中必不可少的一个组成部分，通过 FENCE 设备可以避免因出现不可预知的情况而造成的“脑裂”现象，FENCE 设备的出现，就是为了解决类似这些问题，Fence 设备主要就是通过服务器或存储本身的硬件管理接口，或者外部电源管理设备，来对服务器或存储直接发出硬件管理指令，将服务器重启或关机，或者与网络断开连接。

FENCE 的工作原理是：当意外原因导致主机异常或者宕机时，备机会首先调用 FENCE 设备，然后通过 FENCE 设备将异常主机重启或者从网络隔离，当 FENCE 操作成功执行后，返回信息给备机，备机在接到 FENCE 成功的信息后，开始接管主机的服务和资源。这样通过 FENCE 设备，将异常节点占据的资源进行了释放，保证了资源和服务始终运行在一个节点上。

RHCS 的 FENCE 设备可以分为两种：内部 FENCE 和外部 FENCE，常用的内部 FENCE 有 IBM RSAII 卡，HP 的 iLO 卡，还有 IPMI 的设备等，外部 fence 设备有 UPS、SAN SWITCH、NETWORK SWITCH 等

#### 5、高可用服务管理器

高可用性服务管理主要用来监督、启动和停止集群的应用、服务和资源。它提供了一种对集群服务的管理能力，当一个节点的服务失败时，高可用性集群服务管理进程可以将服务从这个失败节点转移到其它健康节点上来，并且这种服务转移能力是自动、透明的。

RHCS 通过 `rgmanager` 来管理集群服务，`rgmanager` 运行在每个集群节点上，在服务器上对

应的进程为 `clurgmgrd`。

在一个 RHCS 集群中，高可用性服务包含集群服务和集群资源两个方面，集群服务其实就是应用服务，例如 `apache`、`mysql` 等，集群资源有很多种，例如一个 IP 地址、一个运行脚本、`ext3/GFS` 文件系统等。

在 RHCS 集群中，高可用性服务是和一个失败转移域结合在一起的，所谓失败转移域是一个运行特定服务的集群节点的集合。在失败转移域中，可以给每个节点设置相应的优先级，通过优先级的高低来决定节点失败时服务转移的先后顺序，如果没有给节点指定优先级，那么集群高可用服务将在任意节点间转移。因此，通过创建失败转移域不但可以设定服务在节点间转移的顺序，而且可以限制某个服务仅在失败转移域指定的节点内进行切换。

## 6、集群配置管理工具

RHCS 提供了多种集群配置和管理工具，常用的有基于 GUI 的 `system-config-cluster`、`Conga` 等，也提供了基于命令行的管理工具。

`system-config-cluster` 是一个用于创建集群和配置集群节点的图形化管理工具，它有集群节点配置和集群管理两个部分组成，分别用于创建集群节点配置文件和维护节点运行状态。一般用在 RHCS 早期的版本中。

`Conga` 是一种新的基于网络的集群配置工具，与 `system-config-cluster` 不同的是，`Conga` 是通过 web 方式来配置和管理集群节点的。`Conga` 有两部分组成，分别是 `luci` 和 `ricci`，`luci` 安装在一台独立的计算机上，用于配置和管理集群，`ricci` 安装在每个集群节点上，`Luci` 通过 `ricci` 和集群中的每个节点进行通信。

RHCS 也提供了一些功能强大的集群命令行管理工具，常用的有 `clustat`、`cman_tool`、`ccs_tool`、`fence_tool`、`clusvcadm` 等，这些命令的用法将在下面讲述。

## 7、 Redhat GFS

GFS 是 RHCS 为集群系统提供的一个存储解决方案，它允许集群多个节点在块级别上共享存储，每个节点通过共享一个存储空间，保证了访问数据的一致性，更切实的说，GFS 是 RHCS 提供的一个集群文件系统，多个节点同时挂载一个文件系统分区，而文件系统数据不受破坏，这是单一的文件系统，例如 EXT3、EXT2 所不能做到的。

为了实现多个节点对于一个文件系统同时读写操作，GFS 使用锁管理器来管理 I/O 操作，当一个写进程操作一个文件时，这个文件就被锁定，此时不允许其它进程进行读写操作，直到这个写进程正常完成才释放锁，只有当锁被释放后，其它读写进程才能对这个文件进行操作，另外，当一个节点在 GFS 文件系统上修改数据后，这种修改操作会通过 RHCS 底层通信机制立即在其它节点上可见。

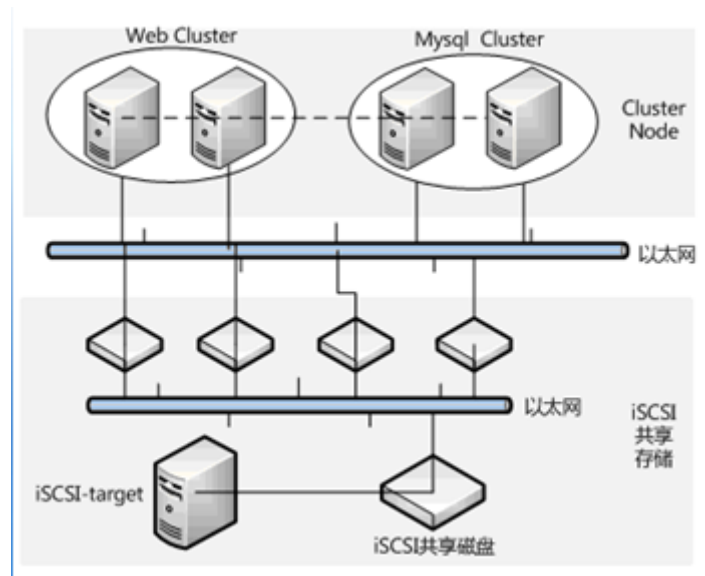
在搭建 RHCS 集群时，GFS 一般作为共享存储，运行在每个节点上，并且可以通过 RHCS 管理工具对 GFS 进行配置和管理。这些需要说明的是 RHCS 和 GFS 之间的关系，一般初学者很容易混淆这个概念：运行 RHCS，GFS 不是必须的，只有在需要共享存储时，才需要 GFS 支持，而搭建 GFS 集群文件系统，必须要有 RHCS 的底层支持，所以安装 GFS 文件系统的节点，必须安装 RHCS 组件。

## 第二篇：实战篇

### 一、 安装环境介绍

这个实例要介绍的是 web+mysql 集群的构建，整个 RHCS 集群共有四台服务器组成，分别由两台主机搭建 web 集群，两台主机搭建 mysql 集群，在这种集群构架下，任何一台 web 服务器故障，都有另一台 web 服务器进行服务接管，同时，任何一台 mysql 服务器故障，也有另一台 mysql 服务器去接管服务，保证了整个应用系统服务的不间断运行。如下图所示：





## 二、安装前准备工作

Centos 是 RHEL 的克隆版本，并且 RHCS 所有功能组件都免费提供，因此下面的讲述以 Centos 为准。

操作系统：统一采用 Centos5.3 版本。为了方便安装 RHCS 套件，在安装操作系统时，建议选择如下这些安装包：

桌面环境：xwindows system、GNOME desktop environment。

开发工具：development tools、x software development、gnome software development、kde software development。

地址规划如下：

主机名	IP 地址	主机用途	虚拟 IP
storgae-server	192.168.12.246	ISCSI 存储端/rhcs 管理端	无
Mysql1	192.168.12.231	Mysql 主用服务器	192.168.12.234
Mysql2	192.168.12.232	Mysql 备用服务器	
web1	192.168.12.230	web 主用服务器	192.168.12.233
web2	192.168.12.240	web 备用服务器	

iSCSI-target 的安装与使用已经在前面文章中做过介绍，不再讲述，这里假定共享的磁盘是/dev/sdb。

### 三、安装 Luci

Luci 是 RHCS 基于 web 的集群配置管理工具，可以从系统光盘找到对应的 Luci 安装包，安装如下：

```
[root@storgae-server ~]#rpm -ivh luci-0.12.2-12.el5.centos.1.i386.rpm
```

安装完成，执行 luci 初始化操作：

```
[root@storgae-server ~]#luci_admin init
```

```
Initializing the Luci server
```

```
Creating the 'admin' user
```

```
Enter password:
```

```
Confirm password:
```

```
Please wait...
```

```
The admin password has been successfully set.
```

```
Generating SSL certificates...
```

```
Luci server has been successfully initialized
```

输入两次密码后，就创建了一个默认登录 luci 的用户 admin。

最后，启动 luci 服务即可：

```
[root@storgae-server ~]# /etc/init.d/luci start
```

服务成功启动后，就可以通过 <https://ip:8084> 访问 luci 了。

为了能让 luci 访问集群其它节点，还需要在/etc/hosts 增加如下内容：

192.168.12.231 Mysql1

192.168.12.232 Mysql2

192.168.12.230 web1

192.168.12.240 web2

到这里为止，在 storgae-server 主机上的设置完成。

#### 四、在集群节点安装 RHCS 软件包

为了保证集群每个节点间可以互相通信，需要将每个节点的主机名信息加入/etc/hosts 文件中，修改完成的/etc/hosts 文件内容如下：

127.0.0.1 localhost

192.168.12.230 web1

192.168.12.240 web2

192.168.12.231 Mysql1

192.168.12.232 Mysql2

将此文件依次复制到集群每个节点的/etc/hosts 文件中。

RHCS 软件包的安装有两种方式，可以通过 luci 管理界面，在创建 Cluster 时，通过在线下载方式自动安装，也可以直接从操作系统光盘找到所需软件包进行手动安装，由于在线安装方式受网络和速度的影响，不建议采用，这里通过手动方式来安装 RHCS 软件包。

安装 RHCS，主要安装的组件包有 cman、gfs2 和 rgmanager，当然在安装这些软件包时可能需要其它依赖的系统包，只需按照提示进行安装即可，下面是一个安装清单，在集群的四个节点分别执行：

```
#install cman
```

```
rpm -ivh perl-XML-Namespacesupport-1.09-1.2.1.noarch.rpm
```

```
rpm -ivh perl-XML-SAX-0.14-8.noarch.rpm
```

```
rpm -ivh perl-XML-LibXML-Common-0.13-8.2.2.i386.rpm
```

```
rpm -ivh perl-XML-LibXML-1.58-6.i386.rpm
```

```
rpm -ivh perl-Net-Telnet-3.03-5.noarch.rpm
```

```
rpm -ivh pexpect-2.3-3.el5.noarch.rpm
```

```
rpm -ivh openais-0.80.6-16.el5_5.2.i386.rpm
```

```
rpm -ivh cman-2.0.115-34.el5.i386.rpm
```

```
#install ricci
```

```
rpm -ivh modcluster-0.12.1-2.el5.centos.i386.rpm
```

```
rpm -ivh ricci-0.12.2-12.el5.centos.1.i386.rpm
```

```
#install gfs2
```

```
rpm -ivh gfs2-utils-0.1.62-20.el5.i386.rpm
```

```
#install rgmanager
```

```
rpm -ivh rgmanager-2.0.52-6.el5.centos.i386.rpm
```

## 五、在集群节点安装配置 iSCSI 客户端

安装 iSCSI 客户端是为了和 iSCSI-target 服务端进行通信，进而将共享磁盘导入到各个集群节点，这里以集群节点 web1 为例，介绍如何安装和配置 iSCSI，剩余其它节点的安装和配置方式与 web1 节点完全相同。

iSCSI 客户端的安装和配置非常简单，只需如下几个步骤即可完成：

```
[root@web1 rhcs]# rpm -ivh iscsi-initiator-utils-6.2.0.871-0.16.el5.i386.rpm
```

```
[root@web1 rhcs]# /etc/init.d/iscsi restart
```

```
[root@web1 rhcs]# iscsiadm -m discovery -t sendtargets -p 192.168.12.246
```

```
[root@web1 rhcs]# /etc/init.d/iscsi restart
```

```
[root@web1 rhcs]# fdisk -l
```

```
Disk /dev/sdb: 10.7 GB, 10737418240 bytes
```

```
64 heads, 32 sectors/track, 10240 cylinders
```

```
Units = cylinders of 2048 * 512 = 1048576 bytes
```

```
Disk /dev/sdb doesn't contain a valid partition table
```

通过 fdisk 的输出可知，/dev/sdb 就是从 iSCSI-target 共享过来的磁盘分区。

至此，安装工作全部结束。

## 六、配置 RHCS 高可用集群

配置 RHCS，其核心就是配置/etc/cluster/cluster.conf 文件，下面通过 web 管理界面介绍如何构造一个 cluster.conf 文件。

在 storgae-server 主机上启动 luci 服务，然后通过浏览器访问

<https://192.168.12.246:8084/>，就可以打开 luci 登录界面，如图 1 所示：



CentOS



CLUSTER AND  
STORAGE SYSTEMS

## 请登录

要访问网站的这部分内容，您需要输入您的用户名和密码。

### 帐号详细信息

#### 登录名

用户名是大小写敏感的，请确认键盘大写锁没有开启。

#### 密码

大小写敏感，请确认大写锁没有开启。

完成后，请注意登出或者退出您的浏览器。

图 1

成功登录后，luci 有三个配置选项，分别是 homebase、cluster 和 storage，其中，cluster 主要用于创建和配置集群系统，storage 用于创建和管理共享存储，而 homebase 主要用于添加、更新、删除 cluster 系统和 storage 设置，同时也可以创建和删除 luci 登录用户。

如图 2 所示：



图 2

### 1、创建一个 cluster

登录 luci 后，切换到 cluster 选项，然后点击左边的 clusters 选框中的 “Create a new cluster”，增加一个 cluster，如图 3 所示：

clusters

Cluster List

Create a New Cluster

Configure

Create a new cluster

Cluster Name

mycluster

Node Hostname	Root Password	Key ID
web1	.....	
web2	.....	
Mysql1	.....	
Mysql2	.....	

Add a cluster node

☐ Download packages

☒ Use locally installed packages.

☐ Enable Shared Storage Support

☐ Reboot nodes before joining cluster

☐ Check if node passwords are identical.

View SSL cert fingerprints

Submit

图 3

在图 3 中，创建的 cluster 名称为 mycluster，“Node Hostname”表示每个节点的主机名称，“Root Password”表示每个节点的 root 用户密码。每个节点的 root 密码可以相同，也可以不同。

在下面的五个选项中，“Download packages”表示在线下载并自动安装 RHCS 软件包，而“Use locally installed packages”表示用本地安装包进行安装，由于 RHCS 组件包在上面的介绍中已经手动安装完成，所以这里选择本地安装即可。剩下的三个复选框分别是启用共享存储支持（Enable Shared Storage Support）、节点加入集群时重启系统（Reboot nodes before joining cluster）和检查节点密码的一致性（Check if node passwords are identical），这些创建 cluster 的设置，可选可不选，这里不做任何选择。



“View SSL cert fingerprints” 用于验证集群各个节点与 luci 通信是否正常，并检测每个节点的配置是否可以创建集群，如果检测失败，会给出相应的错误提示信息。如果验证成功，会输出成功信息。

所有选项填写完成，点击“Submit”进行提交，接下来 luci 开始创建 cluster，如图 4 所示：

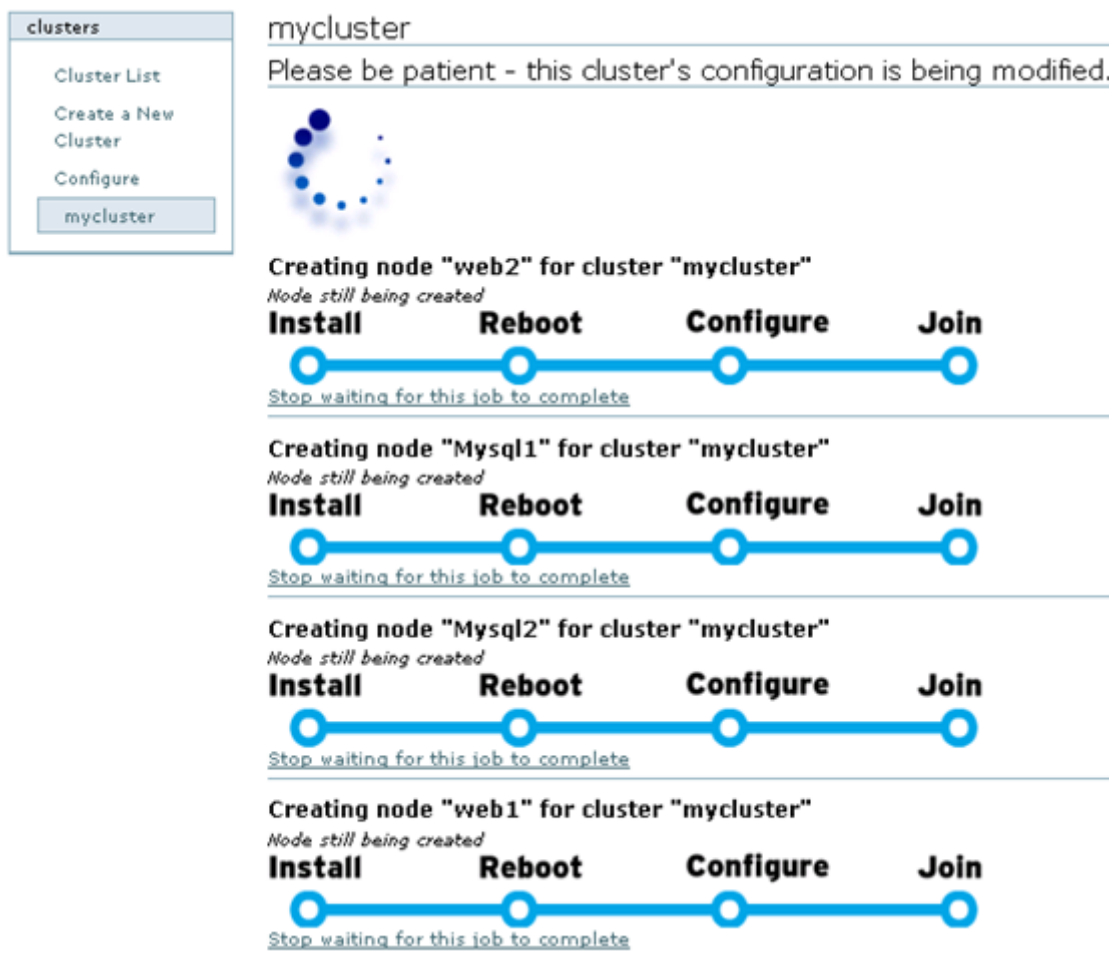


图 4

在经过 Install----Reboot----Configure----Join 四个过程后，如果没有报错，

“mycluster”就创建完成了，其实创建 cluster 的过程，就是 luci 将设定的集群信息写入

到每个集群节点配置文件的过程。Cluster 创建成功后，默认显示“mycluster”的集群全局属性列表，点击 cluster-->Cluster list 来查看创建的 mycluster 的状态，如图 5 所示：

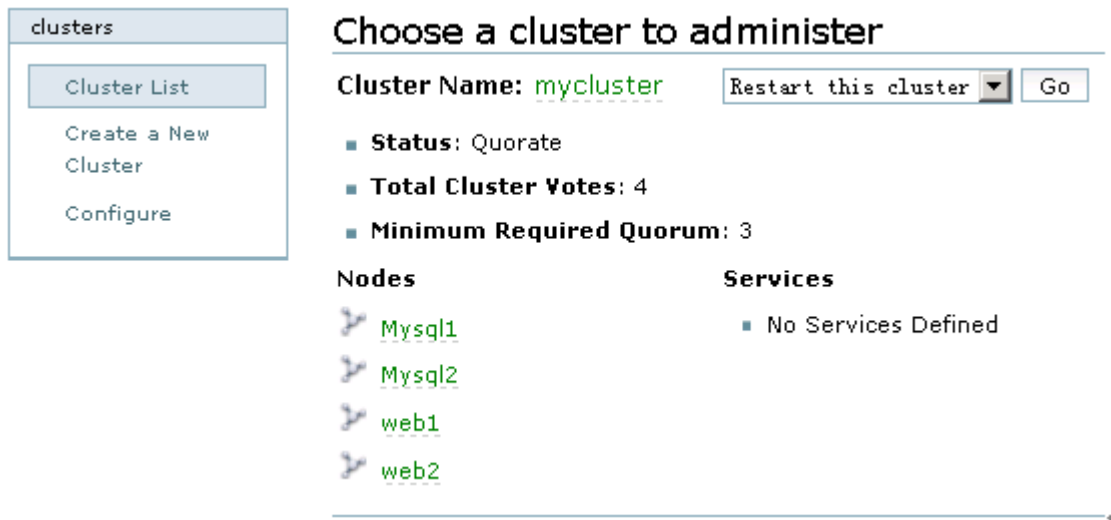


图 5

从图 5 可知，mycluster 集群下有四个节点，正常状态下，节点 Nodes 名称和 Cluster Name 均显示为绿色，如果出现异常，将显示为红色。

点击 Nodes 下面的任意一个节点名称，可以查看此节点的运行状态，如图 6 所示：

clusters

Cluster List

Create a New Cluster

Configure

mycluster

Nodes

Add a Node

Configure

web2

Mysql1

Mysql2

web1


Services

Resources

Failover Domains

Shared Fence Devices

## mycluster

 Node Name: web1

Choose a Task... Go

Status: Cluster member

[Show recent log activity for this node](#)

### Cluster daemons running on this node

Daemon	Currently running	Enabled at start-up
cman	yes	<input checked="" type="checkbox"/>
rgmanager	yes	<input checked="" type="checkbox"/>

Update node daemon properties

### Services on this Node

No cluster services are currently running here

### Failover Domain Membership

This node has no failover domain membership

#### Main Fencing Method

Add a fence device to this level

Update main fence properties

#### Backup Fencing Method

Add a fence device to this level

Update backup fence properties

图 6

从图 6 可以看出，cman 和 rgmanager 服务运行在每个节点上，并且这两个服务需要开机自动启动，它们是 RHCS 的核心守护进程，如果这两个服务在某个节点没有启动，可以通过命令行方式手工启动，命令如下：

```
/etc/init.d/cman start
```

```
/etc/init.d/rgmanager start
```

服务启动成功后，在图 6 中点击“Update node daemon properties”按钮，更新节点的状态。

通过上面的操作，一个简单的 cluster 就创建完成了，但是这个 cluster 目前还是不能工作的，还需要为这个 cluster 创建 Failover Domain、Resources、Service、Shared Fence Device 等，下面依次进行介绍。

## 2、创建 Failover Domain

Failover Domain 是配置集群的失败转移域，通过失败转移域可以将服务和资源的切换限制在指定的节点间，下面的操作将创建两个失败转移域，分别是 webserver-failover 和 mysql-failover。

点击 cluster，然后在 Cluster list 中点击“mycluster”，接着，在左下端的 mycluster 栏中点击 Failover Domains-->Add a Failover Domain，增加一个 Failover Domain，如图 7 所示：

clusters

Cluster List

Create a New Cluster

Configure

mycluster

Nodes

Services

Resources

Failover Domains

Add a Failover Domain

Configure a Failover Domain

Shared Fence Devices

mycluster

Add a Failover Domain

Failover Domain Name

webserver-Failover

Prioritized

☒

Restrict failover to this domain's members

☒

Do not fail back services in this domain

☐

Failover domain membership

Node	Member	Priority
web2	<input checked="" type="checkbox"/>	10
Mysql1	<input type="checkbox"/>	1
Mysql2	<input type="checkbox"/>	1
web1	<input checked="" type="checkbox"/>	1

Submit

图 7

在图 7 中，各个参数的含义如下：

Failover domain name: 创建的失败转移域名称，起一个易记的名字即可。

Prioritized: 是否在 Failover domain 中启用域成员优先级设置，这里选择启用。

Restrict Failover to this domain's member: 表示是否在失败转移域成员中启用服务故障切换限制。这里选择启用。

Do not fail back services in this domain: 表示在这个域中使用故障切回功能，也就是说，主节点故障时，备用节点会自动接管主节点服务和资源，当主节点恢复正常时，集群的服务和资源会从备用节点自动切换到主节点。

然后，在 Failover domain membership 的 Member 复选框中，选择加入此域的节点，这里选择的是 web1 和 web2 节点，然后，在“priority”处将 web1 的优先级设置为 1，web2 的优先级设置为 10。需要说明的是“priority”设置为 1 的节点，优先级是最高的，随着数值的降低，节点优先级也依次降低。

所有设置完成，点击 Submit 按钮，开始创建 Failover domain。

按照上面的介绍，继续添加第二个失败转移域 mysql-failover，在 Failover domain membership 的 Member 复选框中，选择加入此域的节点，这里选择 Mysql1 和 Mysql2 节点，然后，在“priority”处将 Mysql1 的优先级设置为 2，Mysql2 的优先级设置为 8。

### 3、创建 Resources

Resources 是集群的核心，主要包含服务脚本、IP 地址、文件系统等，RHCS 提供的资源如

图 8 所示：

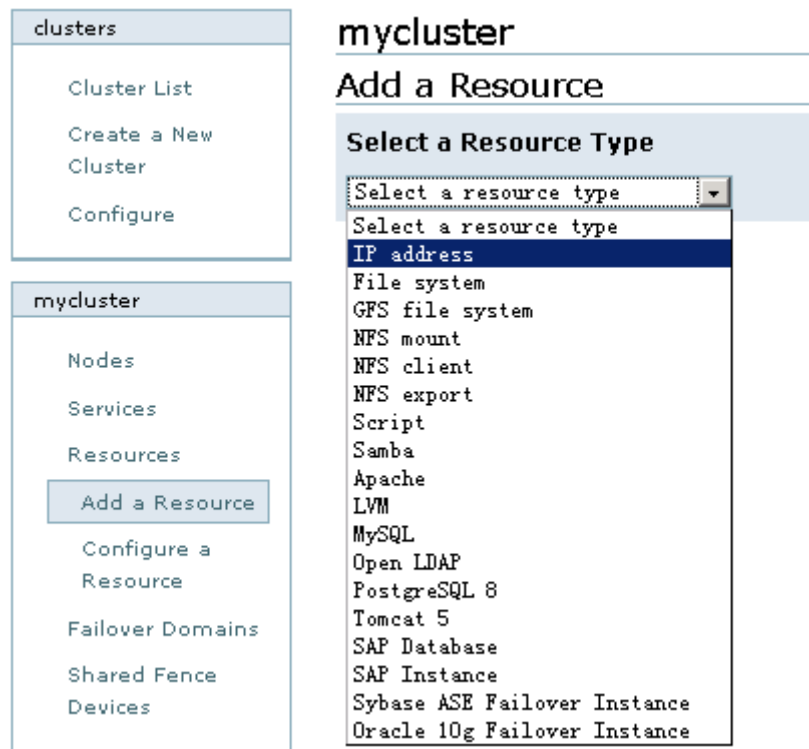


图 8

依次添加 IP 资源、http 服务资源、Mysql 管理脚本资源、ext3 文件系统，如图 9 所示：

clusters

[Cluster List](#)  
[Create a New Cluster](#)  
[Configure](#)

mycluster

[Nodes](#)  
[Services](#)  
[Resources](#)  
[Add a Resource](#)  
[Configure a Resource](#)  
[Failover Domains](#)  
[Shared Fence](#)  
[Devices](#)

mycluster

Resources for mycluster

Resource Name	Type	Configure	Delete
192.168.12.233	IP Address	<a href="#">configure</a>	<a href="#">delete</a>
httpScript	Script	<a href="#">configure</a>	<a href="#">delete</a>
192.168.12.234	IP Address	<a href="#">configure</a>	<a href="#">delete</a>
mysqlscript	Script	<a href="#">configure</a>	<a href="#">delete</a>
ext3-fs	File System	<a href="#">configure</a>	<a href="#">delete</a>

图 9

#### 4、创建 Service

点击 cluster，然后在 Cluster list 中点击“mycluster”，接着，在左下端的 mycluster 栏中点击 Services-->Add a Service，在集群中添加一个服务，如图 10 所示：

**clusters**

- Cluster List
- Create a New Cluster
- Configure

**mycluster**

- Nodes
- Services
  - Add a Service
  - Configure a Service
- Resources
- Failover Domains
- Shared Fence Devices

### mycluster

#### Add a Service

**Service name**

Automatically start this service ☒

Enable NFS lock workarounds ☐

Run exclusive ☐

Failover Domain

Recovery policy

Maximum number of restart failures before relocating

Length of time in seconds after which to forget a restart

#### IP Address Resource Configuration

IP address

Monitor link ☒

This resource is an independent subtree ☐

#### Script Resource Configuration

Name

Full path to script file

This resource is an independent subtree ☐

图 10

所有服务添加完成后，如果应用程序设置正确，服务将自动启动，点击 cluster，然后在 Cluster list 中可以看到两个服务的启动状态，正常情况下，均显示为绿色。如图 11 所示：

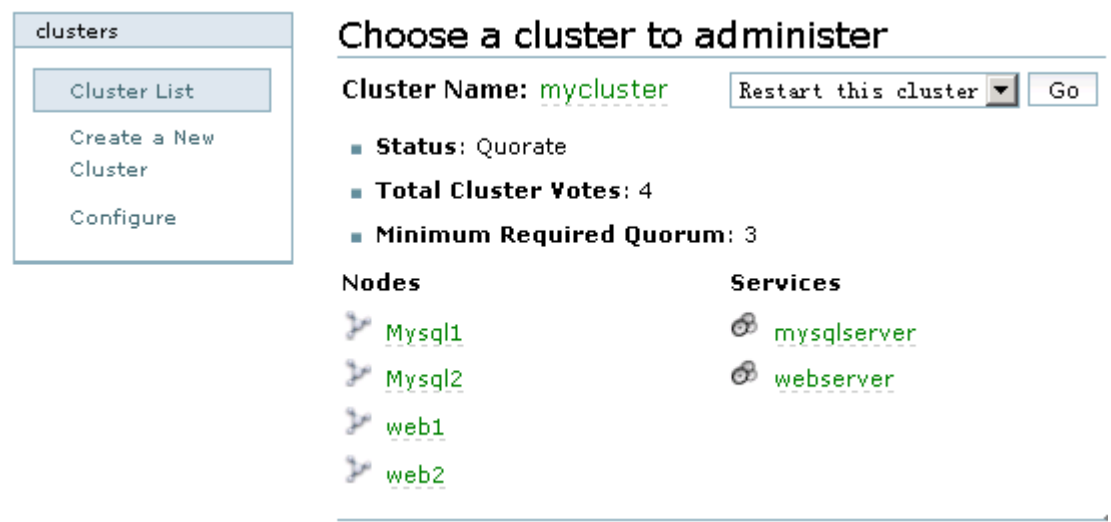


图 11

## 七、配置存储集群 GFS

在上面章节中，我们已经通过 storgae-server 主机将一个磁盘分区共享给了集群系统的四个节点，接下来将进行磁盘分区、格式化、创建文件系统等操作。

### （1）对磁盘进行分区

可以在集群系统任意节点对共享磁盘分区进行磁盘的分区和格式化，这里选择在节点 web1 上进行，首先对共享磁盘进行分区，操作如下：

```
[root@web1 ~]# fdisk /dev/sdb
```

这里将共享磁盘分为三个有效分区，分别将/dev/sdb5 用于 GFS 文件系统，将/dev/sdb6 用于 ext3 文件系统，而将/dev/sdb7 用于表决磁盘，关于表决磁盘，下面马上会进行讲述。

### （2）格式化磁盘

接下来，在 web1 节点将磁盘分区分别格式化为 ext3 和 gfs2 文件系统，操作如下：



```
[root@web1 ~]# mkfs.ext3 /dev/sdb6
```

```
[root@web1 ~]# mkfs.gfs2 -p lock_dlm -t mycluster:my-gfs2 -j 4 /dev/sdb5
```

其中：

```
-p lock_dlm
```

定义为 DLM 锁方式，如果不加此参数，当在两个系统中同时挂载此分区时就会像 EXT3 格式一样，两个系统的信息不能同步。

```
-t mycluster:my-gfs2
```

指定 DLM 锁所在的表名称，mycluster 就是 RHCS 集群的名称，必须与 cluster.conf 文件中 Cluster 标签的 name 值相同。

```
-j 4
```

设定 GFS2 文件系统最多支持多少个节点同时挂载，这个值可以通过 gfs2\_jadd 命令在使用中动态调整。

```
/dev/sdb5
```

指定要格式化的分区设备标识。

所有操作完成后，重启集群所有节点，保证划分的磁盘分区能够被所有节点识别。

### （3）挂载磁盘

所有节点重新启动后，就可以挂载文件系统了，依次在集群的每个节点执行如下操作，将共享文件系统挂载到/gfs2 目录下：

```
[root@web1 ~]#mount -t gfs2 /dev/sdb5 /gfs2 -v
```

```
/sbin/mount.gfs2: mount /dev/sdb5 /gfs2
```

```
/sbin/mount.gfs2: parse_opts: opts = "rw"
```

```

/sbin/mount.gfs2:      clear flag 1 for "rw", flags = 0

/sbin/mount.gfs2: parse_opts: flags = 0

/sbin/mount.gfs2: write "join /gfs2 gfs2 lock_dlm mycluster:my-gfs2 rw /dev/sdb5"

//sbin/mount.gfs2: mount(2) ok

/sbin/mount.gfs2: lock_dlm_mount_result: write "mount_result /gfs2 gfs2 0"

/sbin/mount.gfs2: read_proc_mounts: device = "/dev/sdb5"

/sbin/mount.gfs2: read_proc_mounts: opts = "rw,hostdata=jid=3:id=65540:first=0

“

```

通过“-v”参数可以输出挂载 gfs2 文件系统的过程，有助于理解 gfs2 文件系统和问题排查。

为了能让共享文件系统开机自动挂载磁盘，将以下内容添加到每个集群节点的/etc/fstab 文件中。

```

#GFS      MOUNT POINTS

/dev/sdb5      /gfs2                  gfs2      defaults

1 1

```

## 八、配置表决磁盘

### （1）使用表决磁盘的必要性

在一个多节点的 RHCS 集群系统中，一个节点失败后，集群的服务和资源可以自动转移到其它节点上，但是这种转移是有条件的，例如，在一个四节点的集群中，一旦有两个节点发生故障，整个集群系统将会挂起，集群服务也随即停止，而如果配置了存储集群 GFS 文件系统，那么只要有一个节点发生故障，所有节点挂载的 GFS 文件系统将 hung 住。此时共享存储将无法使用，这种情况的出现，对于高可用的集群系统来说是绝对不允许的，解决这种问

题就要通过表决磁盘来实现了。

## （2）表决磁盘运行机制

表决磁盘，即 Quorum Disk，在 RHCS 里简称 qdisk，是基于磁盘的 Cluster 仲裁服务程序，为了解决小规模集群中投票问题，RHCS 引入了 Quorum 机制，Quorum 表示集群法定的节点数，和 Quorum 对应的是 Quorate，Quorate 是一种状态，表示达到法定节点数。在正常状态下，Quorum 的值是每个节点投票值再加上 QDisk 分区的投票值之和。

QDisk 是一个小于 10MB 的共享磁盘分区，Qdiskd 进程运行在集群的所有节点上，通过 Qdiskd 进程，集群节点定期评估自身的健康情况，并且把自身的状态信息写到指定的共享磁盘分区中，同时 Qdiskd 还可以查看其它节点的状态信息，并传递信息给其它节点。

## （3）RHCS 中表决磁盘的概念

和 qdisk 相关的几个工具有 mkdisk、Heuristics。

mkdisk 是一个集群仲裁磁盘工具集，可以用来创建一个 qdisk 共享磁盘也可以查看共享磁盘的状态信息。mkqdisk 操作只能创建 16 个节点的投票空间，因此目前 qdisk 最多可以支持 16 个节点的 RHCS 高可用集群。

有时候仅靠检测 Qdisk 分区来判断节点状态还是不够的，还可以通过应用程序来扩展对节点状态检测的精度，Heuristics 就是这么一个扩充选项，它允许通过第三方应用程序来辅助定位节点状态，常用的有 ping 网关或路由，或者通过脚本程序等，如果试探失败，qdiskd 会认为此节点失败，进而试图重启此节点，以使节点进入正常状态。

## （4）创建一个表决磁盘

在上面章节中，已经划分了多个共享磁盘分区，这里将共享磁盘分区/dev/sdb7 作为 qdisk 分区，下面是创建一个 qdisk 分区：

```
[root@web1 ~]# mkqdisk -c /dev/sdb7 -l myqdisk

[root@web1 ~]# mkqdisk -L          #查看表决磁盘信息
```

(5) 配置 Qdisk

这里通过 Conga 的 web 界面来配置 Qdisk，首先登录 luci，然后点击 cluster，在 Cluster list 中点击 “mycluster”，然后选择 “Quorum Partition” 一项，如图 12 所示：

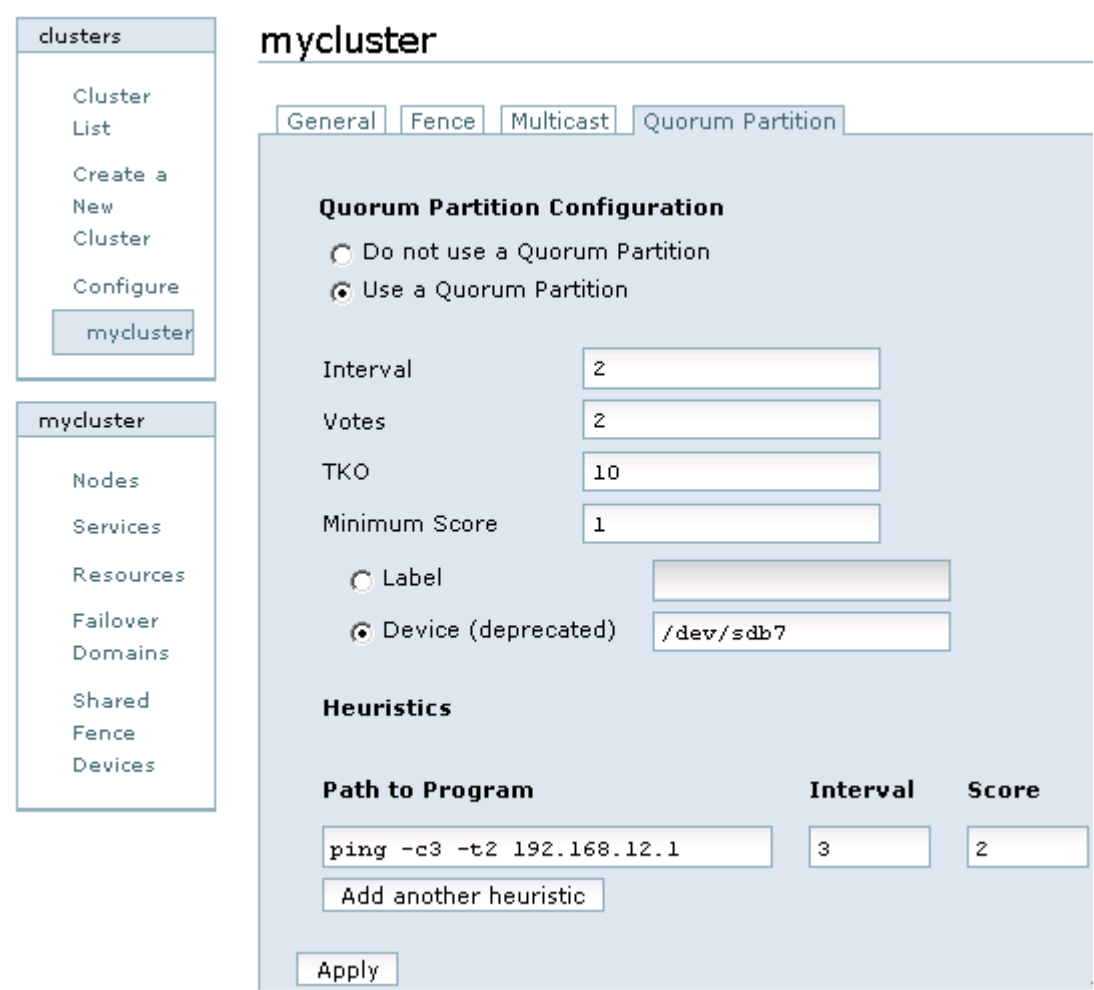


图 12

对图 12 中每个选项的含义解释如下：

Interval：表示间隔多长时间执行一次检查评估，单位是秒。

Votes: 指定 qdisk 分区投票值是多少。

TK0: 表示允许检查失败的次数。一个节点在 TK0\*Interval 时间内如果还连接不上 qdisk 分区，那么就认为此节点失败，会从集群中隔离。

Minimum Score: 指定最小投票值是多少。

Label: Qdisk 分区对应的卷标名，也就是在创建 qdisk 时指定的“myqdisk”，这里建议用卷标名，因为设备名有可能会在系统重启后发生变化，但卷标名称是不会发生改变的。

Device: 指定共享存储在节点中的设备名是什么。

Path to program: 配置第三方应用程序来扩展对节点状态检测的精度，这里配置的是 ping 命令

Score: 设定 ping 命令的投票值。

interval: 设定多长时间执行 ping 命令一次。

## (6) 启动 Qdisk 服务

在集群每个节点执行如下命令，启动 qdiskd 服务：

```
[root@web1 ~]# /etc/init.d/qdiskd start
```

qdiskd 启动后，如果配置正确，qdisk 磁盘将自动进入 online 状态：

```
[root@web1 ~]# clustat -l
```

```
Cluster Status for mycluster @ Sat Aug 21 01:25:40 2010
```

```
Member Status: Quorate
```

Member Name	ID	Status
-----		
Web		1 Online, rgmanager
Mysql1		2 Online, rgmanager

Mysql2	3 Online, rgmanager
web1	4 Online, Local, rgmanager
/dev/sdb7	0 Online, Quorum Disk

至此，Qdisk 已经运行起来了。

## 九、配置 Fence 设备

配置 Fence 设备 是 RHCS 集群系统中必不可少的一个环节,通过 Fence 设备可以防止集群资源（例如文件系统）同时被多个节点占有，保护了共享数据的安全性和一致性节，同时也可以防止节点间脑裂的发生。

GFS 是基于集群底层架构来传递锁信息的，或者说是基于 RHCS 的一种集群文件系统，因此使用 GFS 文件系统也必须要有 fence 设备。

RHCS 提供的 fence device 有两种，一种是内部 fence 设备。常见的有：

IBM 服务器提供的 RSAII 卡

HP 服务器提供的 iLO 卡

DELL 服务器提供的 DRAC 卡

智能平台管理接口 IPMI

常见的外部 fence 设备有：UPS、SAN SWITCH、NETWORK SWITCH，另外如果共享存储是通过 GNBD Server 实现的，那么还可以使用 GNBD 的 fence 功能。

点击 cluster，然后点击“cluster list”中的“mycluster”，在左下角的 mycluster 栏目中选择 Shared Fence Devices—>Add a Sharable Fence Device，在这里选择的 Fence

Device 为“WTI Power Switch”，Fence 的名称为“WTI-Fence”，然后依次输入 IP Address 和 Password，如图 13 所示：

The screenshot displays the RHCS web interface. On the left, there are two navigation menus. The top menu, labeled 'clusters', contains links for 'Cluster List', 'Create a New Cluster', and 'Configure'. The bottom menu, labeled 'mycluster', contains links for 'Nodes', 'Services', 'Resources', 'Failover Domains', 'Shared Fence Devices', 'Add a Fence Device' (highlighted with a blue box), and 'Configure a Fence Device'. The main content area is titled 'mycluster' and 'Add a Sharable Fence Device'. It features a 'Fencing Type' dropdown menu set to 'WTI Power Switch'. Below this, the 'Fence Type' is also listed as 'WTI Power Switch'. The 'Name' field is filled with 'WTI-Fence', the 'IP Address' field with '192.168.12.251', and the 'Password' field with a series of dots. The 'Password Script (optional)' field is empty. At the bottom of the form is a button labeled 'Add this shared fence device'.

图 13

至此，基于 web 界面的 RHCS 配置完成。

## 第三篇：测试篇

集群配置完成后，如何知道集群已经配置成功了呢，下面我们就分情况测试 RHCS 提供的高可用集群和存储集群功能。

### 一、高可用集群测试

在前面的文章中，我们配置了四个节点的集群系统，每个节点的主机名分别是 web1、web2、

Mysql1、Mysql2，四个节点之间的关系是：web1 和 web2 组成 web 集群，运行 webserver 服务，其中 web1 是主节点，正常状态下服务运行在此节点，web2 是备用节点；Mysql1 和 Mysql2 组成 Mysql 集群，运行 mysqlserver 服务，其中，Mysql1 是主节点，正常状态下服务运行在此节点，Mysql2 为备用节点。

下面分四种情况介绍当节点发生宕机时，集群是如何进行切换和工作的。

## 1 节点 web2 宕机时

宕机分为正常关机和异常宕机两种情况，下面分别说明。

### (1) 节点 web2 正常关机

在节点 web2 上执行正常关机命令：

```
[root@web2 ~]#init 0
```

然后在 web1 节点查看/var/log/messages 日志，输出信息如下：

```
Aug 24 00:57:09 web1 clurgmgrd[3321]: <notice> Member 1 shutting down
```

```
Aug 24 00:57:17 web1 qdiskd[2778]: <info> Node 1 shutdown
```

```
Aug 24 00:57:29 web1 openais[2755]: [TOTEM] The token was lost in the OPERATIONAL state.
```

```
Aug 24 00:57:29 web1 openais[2755]: [TOTEM] Receive multicast socket recv buffer size (320000 bytes).
```

```
Aug 24 00:57:29 web1 openais[2755]: [TOTEM] Transmit multicast socket send buffer size (219136 bytes).
```

```
Aug 24 00:57:29 web1 openais[2755]: [TOTEM] entering GATHER state from 2.
```

```
Aug 24 00:57:49 web1 openais[2755]: [TOTEM] entering GATHER state from 0.
```



Aug 24 00:57:49 web1 openais[2755]: [TOTEM] Creating commit token because I am the rep.

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] Saving state aru 73 high seq received 73

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] Storing new sequence id for ring bc8

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] entering COMMIT state.

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] entering RECOVERY state.

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] position [0] member 192.168.12.230:

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] previous ring seq 3012 rep  
192.168.12.230

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] aru 73 high delivered 73 received flag  
1

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] position [1] member 192.168.12.231:

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] previous ring seq 3012 rep  
192.168.12.230

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] aru 73 high delivered 73 received flag  
1

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] position [2] member 192.168.12.232:

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] previous ring seq 3012 rep  
192.168.12.230

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] aru 73 high delivered 73 received flag  
1

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] Did not need to originate any messages  
in recovery.

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] Sending initial ORF token

Aug 24 00:57:49 web1 openais[2755]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 00:57:49 web1 openais[2755]: [CLM ] New Configuration:

Aug 24 00:57:49 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.230)

Aug 24 00:57:49 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.231)

Aug 24 00:57:49 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.232)

Aug 24 00:57:49 web1 openais[2755]: [CLM ] Members Left:

Aug 24 00:57:49 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.240)

Aug 24 00:57:49 web1 kernel: dlm: closing connection to node 1

Aug 24 00:57:49 web1 openais[2755]: [CLM ] Members Joined:

Aug 24 00:57:49 web1 openais[2755]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 00:57:49 web1 openais[2755]: [CLM ] New Configuration:

Aug 24 00:57:49 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.230)

Aug 24 00:57:49 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.231)

Aug 24 00:57:49 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.232)

Aug 24 00:57:49 web1 openais[2755]: [CLM ] Members Left:

Aug 24 00:57:49 web1 openais[2755]: [CLM ] Members Joined:

Aug 24 00:57:49 web1 openais[2755]: [SYNC ] This node is within the primary component  
and will provide service.

Aug 24 00:57:49 web1 openais[2755]: [TOTEM] entering OPERATIONAL state.

```
Aug 24 00:57:49 web1 openais[2755]: [CLM ] got nodejoin message 192.168.12.230
Aug 24 00:57:49 web1 openais[2755]: [CLM ] got nodejoin message 192.168.12.231
Aug 24 00:57:49 web1 openais[2755]: [CLM ] got nodejoin message 192.168.12.232
Aug 24 00:57:49 web1 openais[2755]: [CPG ] got joinlist message from node 3
Aug 24 00:57:49 web1 openais[2755]: [CPG ] got joinlist message from node 4
Aug 24 00:57:49 web1 openais[2755]: [CPG ] got joinlist message from node 2
```

从输出日志可以看出，当 web2 节点正常关机后，qdiskd 进程立刻检测到 web2 节点已经关闭，然后 dlm 锁进程正常关闭了从 web2 的连接，由于是正常关闭节点 web2，所以 RHCS 认为整个集群系统没有发生异常，仅仅把节点 web2 从集群中隔离而已。请重点查看日志中斜体部分。

重新启动 web2 节点，然后在 web1 上继续观察日志信息：

```
Aug 24 01:10:50 web1 openais[2755]: [TOTEM] entering GATHER state from 11.
Aug 24 01:10:51 web1 openais[2755]: [TOTEM] Creating commit token because I am the
rep.
Aug 24 01:10:51 web1 openais[2755]: [TOTEM] Saving state aru 2b high seq received
2b
Aug 24 01:10:51 web1 openais[2755]: [TOTEM] Storing new sequence id for ring bcc
Aug 24 01:10:51 web1 openais[2755]: [TOTEM] entering COMMIT state.
Aug 24 01:10:51 web1 openais[2755]: [TOTEM] entering RECOVERY state.
Aug 24 01:10:51 web1 openais[2755]: [TOTEM] position [0] member 192.168.12.230:
Aug 24 01:10:51 web1 openais[2755]: [TOTEM] previous ring seq 3016 rep
192.168.12.230
```

Aug 24 01:10:51 web1 openais[2755]: [TOTEM] aru 2b high delivered 2b received flag

1

Aug 24 01:10:51 web1 openais[2755]: [TOTEM] position [1] member 192.168.12.231:

Aug 24 01:10:51 web1 openais[2755]: [TOTEM] previous ring seq 3016 rep

192.168.12.230

Aug 24 01:10:51 web1 openais[2755]: [TOTEM] aru 2b high delivered 2b received flag

1

Aug 24 01:10:51 web1 openais[2755]: [TOTEM] position [2] member 192.168.12.232:

Aug 24 01:10:51 web1 openais[2755]: [TOTEM] previous ring seq 3016 rep

192.168.12.230

Aug 24 01:10:51 web1 openais[2755]: [TOTEM] aru 2b high delivered 2b received flag

1

Aug 24 01:10:51 web1 openais[2755]: [TOTEM] position [3] member 192.168.12.240:

Aug 24 01:10:51 web1 openais[2755]: [TOTEM] previous ring seq 3016 rep

192.168.12.240

Aug 24 01:10:51 web1 openais[2755]: [TOTEM] aru c high delivered c received flag

1

Aug 24 01:10:51 web1 openais[2755]: [TOTEM] Did not need to originate any messages  
in recovery.

Aug 24 01:10:51 web1 openais[2755]: [TOTEM] Sending initial ORF token

Aug 24 01:10:51 web1 openais[2755]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 01:10:51 web1 openais[2755]: [CLM ] New Configuration:

Aug 24 01:10:51 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.230)

Aug 24 01:10:51 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.231)

Aug 24 01:10:51 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.232)

Aug 24 01:10:51 web1 openais[2755]: [CLM ] Members Left:

Aug 24 01:10:51 web1 openais[2755]: [CLM ] Members Joined:

Aug 24 01:10:51 web1 openais[2755]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 01:10:51 web1 openais[2755]: [CLM ] New Configuration:

Aug 24 01:10:51 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.230)

Aug 24 01:10:51 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.231)

Aug 24 01:10:51 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.232)

Aug 24 01:10:51 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.240)

Aug 24 01:10:51 web1 openais[2755]: [CLM ] Members Left:

**Aug 24 01:10:51 web1 openais[2755]: [CLM ] Members Joined:**

**Aug 24 01:10:51 web1 openais[2755]: [CLM ] r(0) ip(192.168.12.240)**

Aug 24 01:10:51 web1 openais[2755]: [SYNC ] This node is within the primary component and will provide service.

Aug 24 01:10:51 web1 openais[2755]: [TOTEM] entering OPERATIONAL state.

Aug 24 01:10:51 web1 openais[2755]: [CLM ] got nodejoin message 192.168.12.230

Aug 24 01:10:51 web1 openais[2755]: [CLM ] got nodejoin message 192.168.12.231

Aug 24 01:10:51 web1 openais[2755]: [CLM ] got nodejoin message 192.168.12.232

**Aug 24 01:10:51 web1 openais[2755]: [CLM ] got nodejoin message 192.168.12.240**

Aug 24 01:10:51 web1 openais[2755]: [CPG ] got joinlist message from node 3

Aug 24 01:10:51 web1 openais[2755]: [CPG ] got joinlist message from node 4

Aug 24 01:10:51 web1 openais[2755]: [CPG ] got joinlist message from node 2

**Aug 24 01:10:55 web1 kernel: dlm: connecting to 1**

从输出可知，重新启动节点 web2 后，openais 底层通信进程检测到 web2 节点已经激活，并且将 web2 重新加入集群中，请重点查看日志中斜体部分。

## (2) 节点 web2 异常宕机

在节点 web2 上执行如下命令，让内核崩溃：

```
[root@web2 ~]#echo c>/proc/sysrq-trigger
```

然后在节点 Mysql1 上查看/var/log/messages 日志，信息如下：

Aug 24 02:26:16 Mysql1 openais[2649]: [TOTEM] entering GATHER state from 12.

**Aug 24 02:26:28 Mysql1 qdiskd[2672]: <notice> Node 1 evicted**

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] entering GATHER state from 11.

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] Saving state aru 78 high seq received  
78

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] Storing new sequence id for ring bd0

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] entering COMMIT state.

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] entering RECOVERY state.

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] position [0] member 192.168.12.230:

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] previous ring seq 3020 rep  
192.168.12.230

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] aru 78 high delivered 78 received flag

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] position [1] member 192.168.12.231:

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] previous ring seq 3020 rep

192.168.12.230

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] aru 78 high delivered 78 received flag

1

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] position [2] member 192.168.12.232:

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] previous ring seq 3020 rep

192.168.12.230

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] aru 78 high delivered 78 received flag

1

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] Did not need to originate any messages  
in recovery.

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] New Configuration:

**Aug 24 02:26:36 Mysql1 kernel: dlm: closing connection to node 1**

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] r(0) ip(192.168.12.230)

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] r(0) ip(192.168.12.231)

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] r(0) ip(192.168.12.232)

**Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] Members Left:**

**Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] r(0) ip(192.168.12.240)**

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] Members Joined:

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] New Configuration:

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] r(0) ip(192.168.12.230)

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] r(0) ip(192.168.12.231)

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] r(0) ip(192.168.12.232)

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] Members Left:

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] Members Joined:

Aug 24 02:26:36 Mysql1 openais[2649]: [SYNC ] This node is within the primary component and will provide service.

Aug 24 02:26:36 Mysql1 fenced[2688]: web2 not a cluster member after 0 sec post\_fail\_delay

Aug 24 02:26:36 Mysql1 openais[2649]: [TOTEM] entering OPERATIONAL state.

**Aug 24 02:26:36 Mysql1 fenced[2688]: fencing node "web2"**

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] got nodejoin message 192.168.12.230

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] got nodejoin message 192.168.12.231

Aug 24 02:26:36 Mysql1 openais[2649]: [CLM ] got nodejoin message 192.168.12.232

Aug 24 02:26:36 Mysql1 openais[2649]: [CPG ] got joinlist message from node 3

Aug 24 02:26:36 Mysql1 openais[2649]: [CPG ] got joinlist message from node 4

Aug 24 02:26:36 Mysql1 openais[2649]: [CPG ] got joinlist message from node 2

**Aug 24 02:26:45 Mysql1 fenced[2688]: fence "web2" success**

Aug 24 02:26:45 Mysql1 kernel: GFS2: fsid=mycluster:my-gfs2.2: jid=3: Trying to acquire journal lock...

Aug 24 02:26:45 Mysql1 kernel: GFS2: fsid=mycluster:my-gfs2.2: jid=3: Looking at



journal...

Aug 24 02:26:45 Mysql1 kernel: GFS2: fsid=mycluster:my-gfs2.2: jid=3: Done

从输出信息可知，qdiskd 首先检测到 web2 出现异常，然后将它从集群中隔离，由于是异常宕机，所以 RHCS 为了保证集群资源的唯一性，必须重置 web2 节点，于是 fenced 进程启动，在 fenced 进程没有返回成功信息之前，所有节点挂载的 GFS2 共享分区将无法使用，处于 hung 住的状态。直到 fence 成功。

此时，在节点 web1 上查看 web2 的集群状态：

```
[root@web1 ~]# clustat -m web2
```

Member Name	ID	Status
web2	1	Offline

由输出可知，web2 已经处于 offline 状态了。

## 2 节点 Mysql2 宕机时

节点 Mysql2 在正常关机和异常宕机时，RHCS 的切换状态与上面讲述的 web2 节点情况一模一样，这么不在重复讲述。

## 3 节点 web1 宕机时

### (1) 节点 web1 正常关机

```
[root@web1 ~]# init 0
```

然后在节点 web2 上查看 /var/log/messages 日志，信息如下：

Aug 24 02:06:13 web2 last message repeated 3 times

Aug 24 02:14:58 web2 clurgmgrd[3239]: <notice> Member 4 shutting down

Aug 24 02:15:03 web2 clurgmgrd[3239]: <notice> Starting stopped service  
service:webserver

Aug 24 02:15:05 web2 avahi-daemon[3110]: Registering new address record for  
192.168.12.233 on eth0.

Aug 24 02:15:06 web2 in.rdiscd[4451]: setsockopt (IP\_ADD\_MEMBERSHIP): Address  
already in use

Aug 24 02:15:06 web2 in.rdiscd[4451]: Failed joining addresses

Aug 24 02:15:07 web2 clurgmgrd[3239]: <notice> Service service:webserver started

Aug 24 02:15:08 web2 qdiskd[2712]: <info> Node 4 shutdown

Aug 24 02:15:21 web2 openais[2689]: [TOTEM] entering GATHER state from 12.

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] entering GATHER state from 11.

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] Saving state aru b7 high seq received  
b7

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] Storing new sequence id for ring bd8

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] entering COMMIT state.

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] entering RECOVERY state.

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] position [0] member 192.168.12.231:

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] previous ring seq 3028 rep  
192.168.12.230

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] aru b7 high delivered b7 received flag  
1

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] position [1] member 192.168.12.232:

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] previous ring seq 3028 rep

192.168.12.230

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] aru b7 high delivered b7 received flag

1

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] position [2] member 192.168.12.240:

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] previous ring seq 3028 rep

192.168.12.230

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] aru b7 high delivered b7 received flag

1

Aug 24 02:15:41 web2 openais[2689]: [TOTEM] Did not need to originate any messages  
in recovery.

Aug 24 02:15:41 web2 openais[2689]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 02:15:41 web2 openais[2689]: [CLM ] New Configuration:

Aug 24 02:15:41 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.231)

Aug 24 02:15:41 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.232)

Aug 24 02:15:41 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.240)

Aug 24 02:15:41 web2 openais[2689]: [CLM ] Members Left:

**Aug 24 02:15:41 web2 kernel: dlm: closing connection to node 4**

**Aug 24 02:15:41 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.230)**

Aug 24 02:15:41 web2 openais[2689]: [CLM ] Members Joined:

Aug 24 02:15:41 web2 openais[2689]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 02:15:41 web2 openais[2689]: [CLM ] New Configuration:

```

Aug 24 02:15:41 web2 openais[2689]: [CLM  ]          r(0) ip(192.168.12.231)
Aug 24 02:15:41 web2 openais[2689]: [CLM  ]          r(0) ip(192.168.12.232)
Aug 24 02:15:41 web2 openais[2689]: [CLM  ]          r(0) ip(192.168.12.240)
Aug 24 02:15:41 web2 openais[2689]: [CLM  ] Members Left:
Aug 24 02:15:41 web2 openais[2689]: [CLM  ] Members Joined:
Aug 24 02:15:41 web2 openais[2689]: [SYNC ] This node is within the primary component
and will provide service.
Aug 24 02:15:41 web2 openais[2689]: [TOTEM] entering OPERATIONAL state.
Aug 24 02:15:41 web2 openais[2689]: [CLM  ] got nodejoin message 192.168.12.231
Aug 24 02:15:41 web2 openais[2689]: [CLM  ] got nodejoin message 192.168.12.232
Aug 24 02:15:41 web2 openais[2689]: [CLM  ] got nodejoin message 192.168.12.240
Aug 24 02:15:41 web2 openais[2689]: [CPG  ] got joinlist message from node 2
Aug 24 02:15:41 web2 openais[2689]: [CPG  ] got joinlist message from node 3
Aug 24 02:15:41 web2 openais[2689]: [CPG  ] got joinlist message from node 1

```

从输出日志可知,节点 web1 正常关机后,节点 web1 的服务和 IP 资源自动切换到了节点 web2 上,然后由 qdiskd 进程将节点 web1 从集群系统中隔离。由于 web1 节点是正常关闭,所以集群中 GFS2 共享文件系统可以正常读写,不受 web1 关闭的影响。

此时,在 web2 查看节点 web1 的状态:

```
[root@web2 ~]# clustat -m web1
```

Member Name	ID	Status
-----	-----	-----
web1	4	Offline

从输出可知，web1 节点已经处于 offline 状态了。

接着，登录到节点 web2，查看集群服务和 IP 资源是否正常切换，操作如下：

```
[root@web2 ~]# clustat -s webserver
```

Service Name	Owner (Last)	State
-----	-----	-----
service:webserver	web2	started

```
[root@web2 ~]# ip addr show|grep eth0
```

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    inet 192.168.12.240/24 brd 192.168.12.255 scope global eth0
    inet 192.168.12.233/24 scope global secondary eth0
```

从输出可知，集群服务和 IP 地址已经成功切换到 web2 节点。

最后，重新启动节点 web1，然后在节点 web2 查看/var/log/messages 日志，信息如下：

```
Aug 24 02:42:36 web2 openais[2689]: [TOTEM] entering GATHER state from 11.
Aug 24 02:42:36 web2 openais[2689]: [TOTEM] Saving state aru 2b high seq received
2b
Aug 24 02:42:36 web2 openais[2689]: [TOTEM] Storing new sequence id for ring bdc
Aug 24 02:42:36 web2 openais[2689]: [TOTEM] entering COMMIT state.
Aug 24 02:42:36 web2 openais[2689]: [TOTEM] entering RECOVERY state.
Aug 24 02:42:36 web2 openais[2689]: [TOTEM] position [0] member 192.168.12.230:
Aug 24 02:42:36 web2 openais[2689]: [TOTEM] previous ring seq 3028 rep
192.168.12.230
```

Aug 24 02:42:36 web2 openais[2689]: [TOTEM] aru 0 high delivered 0 received flag

1

Aug 24 02:42:36 web2 openais[2689]: [TOTEM] position [1] member 192.168.12.231:

Aug 24 02:42:36 web2 openais[2689]: [TOTEM] previous ring seq 3032 rep

192.168.12.231

Aug 24 02:42:36 web2 openais[2689]: [TOTEM] aru 2b high delivered 2b received flag

1

Aug 24 02:42:36 web2 openais[2689]: [TOTEM] position [2] member 192.168.12.232:

Aug 24 02:42:36 web2 openais[2689]: [TOTEM] previous ring seq 3032 rep

192.168.12.231

Aug 24 02:42:36 web2 openais[2689]: [TOTEM] aru 2b high delivered 2b received flag

1

Aug 24 02:42:36 web2 openais[2689]: [TOTEM] position [3] member 192.168.12.240:

Aug 24 02:42:36 web2 openais[2689]: [TOTEM] previous ring seq 3032 rep

192.168.12.231

Aug 24 02:42:36 web2 openais[2689]: [TOTEM] aru 2b high delivered 2b received flag

1

Aug 24 02:42:36 web2 openais[2689]: [TOTEM] Did not need to originate any messages  
in recovery.

Aug 24 02:42:36 web2 openais[2689]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 02:42:36 web2 openais[2689]: [CLM ] New Configuration:

Aug 24 02:42:36 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.231)

Aug 24 02:42:36 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.232)

Aug 24 02:42:36 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.240)

Aug 24 02:42:36 web2 openais[2689]: [CLM ] Members Left:

Aug 24 02:42:36 web2 openais[2689]: [CLM ] Members Joined:

Aug 24 02:42:36 web2 openais[2689]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 02:42:36 web2 openais[2689]: [CLM ] New Configuration:

Aug 24 02:42:36 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.230)

Aug 24 02:42:36 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.231)

Aug 24 02:42:36 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.232)

Aug 24 02:42:36 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.240)

Aug 24 02:42:36 web2 openais[2689]: [CLM ] Members Left:

**Aug 24 02:42:36 web2 openais[2689]: [CLM ] Members Joined:**

**Aug 24 02:42:36 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.230)**

Aug 24 02:42:36 web2 openais[2689]: [SYNC ] This node is within the primary component and will provide service.

Aug 24 02:42:36 web2 openais[2689]: [TOTEM] entering OPERATIONAL state.

Aug 24 02:42:36 web2 openais[2689]: [CLM ] got nodejoin message 192.168.12.230

Aug 24 02:42:36 web2 openais[2689]: [CLM ] got nodejoin message 192.168.12.231

Aug 24 02:42:36 web2 openais[2689]: [CLM ] got nodejoin message 192.168.12.232

Aug 24 02:42:36 web2 openais[2689]: [CLM ] got nodejoin message 192.168.12.240

Aug 24 02:42:36 web2 openais[2689]: [CPG ] got joinlist message from node 3

Aug 24 02:42:36 web2 openais[2689]: [CPG ] got joinlist message from node 1

Aug 24 02:42:36 web2 openais[2689]: [CPG ] got joinlist message from node 2

Aug 24 02:42:40 web2 kernel: dlm: got connection from 4

Aug 24 02:43:06 web2 clurgmgrd[3239]: <notice> Relocating service:webserver to better node web1

Aug 24 02:43:06 web2 clurgmgrd[3239]: <notice> Stopping service service:webserver

Aug 24 02:43:07 web2 avahi-daemon[3110]: Withdrawing address record for 192.168.12.233 on eth0.

Aug 24 02:43:17 web2 clurgmgrd[3239]: <notice> Service service:webserver is stopped

从输出可知，节点 web1 在重新启动后，再次被加入到集群系统中，同时停止自身的服务以及释放 IP 资源。这个切换方式跟集群设置的 Failover Domain 策略有关，在创建的的失败转移域 webserver-Failover 中，没有加入 “Do not fail back services in this domain” 一项功能，也就是主节点在重新启动后，自动将服务切换回来。

此时在节点 web1 查看 /var/log/messages 日志，信息如下：

Aug 24 02:43:19 web1 clurgmgrd[3252]: <notice> stop on script "mysqlscript" returned 5 (program not installed)

Aug 24 02:43:35 web1 clurgmgrd[3252]: <notice> Starting stopped service service:webserver

Aug 24 02:43:37 web1 avahi-daemon[3126]: Registering new address record for 192.168.12.233 on eth0.

Aug 24 02:43:38 web1 in.rdiscd[4075]: setsockopt (IP\_ADD\_MEMBERSHIP): Address already in use

Aug 24 02:43:38 web1 in.rdiscd[4075]: Failed joining addresses



**Aug 24 02:43:39 web1 clurgmgrd[3252]: <notice> Service service:webserver started**

这个输出表明，web1 在重启后，自动将集群服务和 IP 资源切换回来。

## (2) 节点 web1 异常宕机

在节点 web1 上执行如下命令，让系统内核崩溃：

```
[root@web2 ~]#echo c>/proc/sysrq-trigger
```

然后在节点 web2 上查看/var/log/messages 日志，信息如下：

Aug 24 02:59:57 web2 openais[2689]: [TOTEM] entering GATHER state from 12.

**Aug 24 03:00:10 web2 qdiskd[2712]: <notice> Node 4 evicted**

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] entering GATHER state from 11.

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] Saving state aru 92 high seq received  
92

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] Storing new sequence id for ring be0

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] entering COMMIT state.

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] entering RECOVERY state.

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] position [0] member 192.168.12.231:

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] previous ring seq 3036 rep  
192.168.12.230

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] aru 92 high delivered 92 received flag  
1

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] position [1] member 192.168.12.232:

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] previous ring seq 3036 rep  
192.168.12.230

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] aru 92 high delivered 92 received flag

1

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] position [2] member 192.168.12.240:

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] previous ring seq 3036 rep

192.168.12.230

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] aru 92 high delivered 92 received flag

1

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] Did not need to originate any messages  
in recovery.

Aug 24 03:00:17 web2 openais[2689]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 03:00:17 web2 openais[2689]: [CLM ] New Configuration:

Aug 24 03:00:17 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.231)

Aug 24 03:00:17 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.232)

Aug 24 03:00:17 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.240)

Aug 24 03:00:17 web2 openais[2689]: [CLM ] Members Left:

Aug 24 03:00:17 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.230)

Aug 24 03:00:17 web2 kernel: dlm: closing connection to node 4

Aug 24 03:00:17 web2 openais[2689]: [CLM ] Members Joined:

Aug 24 03:00:17 web2 openais[2689]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 03:00:17 web2 openais[2689]: [CLM ] New Configuration:

Aug 24 03:00:17 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.231)

Aug 24 03:00:17 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.232)

Aug 24 03:00:17 web2 openais[2689]: [CLM ] r(0) ip(192.168.12.240)

Aug 24 03:00:17 web2 openais[2689]: [CLM ] Members Left:

Aug 24 03:00:17 web2 openais[2689]: [CLM ] Members Joined:

Aug 24 03:00:17 web2 openais[2689]: [SYNC ] This node is within the primary component and will provide service.

Aug 24 03:00:17 web2 openais[2689]: [TOTEM] entering OPERATIONAL state.

Aug 24 03:00:17 web2 openais[2689]: [CLM ] got nodejoin message 192.168.12.231

Aug 24 03:00:17 web2 openais[2689]: [CLM ] got nodejoin message 192.168.12.232

Aug 24 03:00:17 web2 openais[2689]: [CLM ] got nodejoin message 192.168.12.240

Aug 24 03:00:17 web2 openais[2689]: [CPG ] got joinlist message from node 2

Aug 24 03:00:17 web2 fenced[2728]: web1 not a cluster member after 0 sec post\_fail\_delay

Aug 24 03:00:17 web2 openais[2689]: [CPG ] got joinlist message from node 3

Aug 24 03:00:17 web2 fenced[2728]: fencing node "web1"

Aug 24 03:00:17 web2 openais[2689]: [CPG ] got joinlist message from node 1

Aug 24 03:00:55 web2 fenced[2728]: fence "web1" success

Aug 24 03:00:55 web2 kernel: GFS2: fsid=mycluster:my-gfs2.3: jid=0: Trying to acquire journal lock...

Aug 24 03:00:55 web2 kernel: GFS2: fsid=mycluster:my-gfs2.3: jid=0: Looking at journal...

Aug 24 03:00:55 web2 kernel: GFS2: fsid=mycluster:my-gfs2.3: jid=0: Done

Aug 24 03:00:55 web2 clurgmgrd[3239]: <notice> Taking over service

```
service:webserver from down member web1
```

```
Aug 24 03:00:55 web2 avahi-daemon[3110]: Registering new address record for  
192.168.12.233 on eth0.
```

```
Aug 24 03:00:55 web2 clurgmgrd[3239]: <notice> Service service:webserver started
```

从输出日志可以看出，web1 在异常宕机后，首先由 qdiskd 进程将失败节点从集群中隔离，然后 qdiskd 进程将结果返回给 cman 进程，cman 进程接着去调用 Fence 进程，最后 Fence 进程根据事先设置好的 Fence agent 调用 Fence 设备将 web1 成功 Fence 掉，clurgmgrd 进程在接到成功 Fence 的信息后，web2 开始接管 web1 的服务和 IP 资源，同时释放 dlm 锁，GFS2 文件系统可以正常读写。

#### 4 节点 Mysql1 宕机时

节点 Mysql1 在正常关机和异常宕机时，RHCS 的切换状态与上面讲述的 web1 节点宕机情况一模一样，这么不在重复演示。

#### 5 四个节点任意三个宕机

这里将 web1、Mysql2、Mysql1 依次异常宕机，然后在 web2 节点查看 RHCS 是如何进行切换动作的。

首先停止 web1 节点，查看/var/log/messages 日志，信息如下：

```
Aug 24 18:57:55 web2 openais[2691]: [TOTEM] entering GATHER state from 12.
```

```
Aug 24 18:58:14 web2 qdiskd[2714]: <notice> Writing eviction notice for node 4
```

```
Aug 24 18:58:15 web2 openais[2691]: [TOTEM] entering GATHER state from 0.
```

```
Aug 24 18:58:15 web2 openais[2691]: [TOTEM] Saving state aru 8e high seq received  
8e
```

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] Storing new sequence id for ring c14

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] entering COMMIT state.

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] entering RECOVERY state.

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] position [0] member 192.168.12.231:

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] previous ring seq 3088 rep

192.168.12.230

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] aru 8e high delivered 8e received flag

1

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] position [1] member 192.168.12.232:

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] previous ring seq 3088 rep

192.168.12.230

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] aru 8e high delivered 8e received flag

1

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] position [2] member 192.168.12.240:

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] previous ring seq 3088 rep

192.168.12.230

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] aru 8e high delivered 8e received flag

1

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] Did not need to originate any messages  
in recovery.

Aug 24 18:58:15 web2 openais[2691]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 18:58:15 web2 openais[2691]: [CLM ] New Configuration:

Aug 24 18:58:15 web2 openais[2691]: [CLM ] r(0) ip(192.168.12.231)

Aug 24 18:58:15 web2 openais[2691]: [CLM ] r(0) ip(192.168.12.232)

Aug 24 18:58:15 web2 openais[2691]: [CLM ] r(0) ip(192.168.12.240)

Aug 24 18:58:15 web2 kernel: dlm: closing connection to node 4

Aug 24 18:58:15 web2 openais[2691]: [CLM ] Members Left:

Aug 24 18:58:15 web2 openais[2691]: [CLM ] r(0) ip(192.168.12.230)

Aug 24 18:58:15 web2 openais[2691]: [CLM ] Members Joined:

Aug 24 18:58:15 web2 openais[2691]: [CLM ] CLM CONFIGURATION CHANGE

Aug 24 18:58:15 web2 openais[2691]: [CLM ] New Configuration:

Aug 24 18:58:15 web2 openais[2691]: [CLM ] r(0) ip(192.168.12.231)

Aug 24 18:58:15 web2 openais[2691]: [CLM ] r(0) ip(192.168.12.232)

Aug 24 18:58:15 web2 openais[2691]: [CLM ] r(0) ip(192.168.12.240)

**Aug 24 18:58:15 web2 openais[2691]: [CLM ] Members Left:**

Aug 24 18:58:15 web2 openais[2691]: [CLM ] Members Joined:

Aug 24 18:58:15 web2 openais[2691]: [SYNC ] This node is within the primary component and will provide service.

Aug 24 18:58:15 web2 openais[2691]: [TOTEM] entering OPERATIONAL state.

Aug 24 18:58:15 web2 openais[2691]: [CLM ] got nodejoin message 192.168.12.231

Aug 24 18:58:15 web2 openais[2691]: [CLM ] got nodejoin message 192.168.12.232

Aug 24 18:58:15 web2 openais[2691]: [CLM ] got nodejoin message 192.168.12.240

**Aug 24 18:58:15 web2 fenced[2730]: web1 not a cluster member after 0 sec**

**post\_fail\_delay**

```
Aug 24 18:58:15 web2 openais[2691]: [CPG   ] got joinlist message from node 3

Aug 24 18:58:15 web2 fenced[2730]: fencing node "web1"

Aug 24 18:58:15 web2 openais[2691]: [CPG   ] got joinlist message from node 1

Aug 24 18:58:15 web2 openais[2691]: [CPG   ] got joinlist message from node 2

Aug 24 18:58:17 web2 qdiskd[2714]: <notice> Node 4 evicted

Aug 24 18:58:29 web2 fenced[2730]: fence "web1" success

Aug 24 18:58:29 web2 kernel: GFS2: fsid=mycluster:my-gfs2.1: jid=3: Trying to
acquire journal lock...

Aug 24 18:58:29 web2 kernel: GFS2: fsid=mycluster:my-gfs2.1: jid=3: Looking at
journal...

Aug 24 18:58:29 web2 kernel: GFS2: fsid=mycluster:my-gfs2.1: jid=3: Done

Aug 24 18:58:30 web2 clurgmgrd[3300]: <notice> Taking over service
service:webserver from down member web1

Aug 24 18:58:32 web2 avahi-daemon[3174]: Registering new address record for
192.168.12.233 on eth0.

Aug 24 18:58:33 web2 clurgmgrd[3300]: <notice> Service service:webserver started
```

通过观察日志可以发现，qdiskd 进程会首先将节点 web1 从集群隔离，然后由 fenced 进程将 web1 成功 fence 掉，最后，web2 才接管了 web1 的服务和 IP 资源。这里有个先后问题，也就是说，只有 Fence 成功，集群资源切换才会进行。

如果 fenced 进程没有成功将 web1 节点 fence 掉，那么 RHCS 会进入等待状态，此时集群系统 GFS2 共享存储也将变得不可读写，直到 Fence 进程返回成功信息，集群才开始进行资源切换，同时 GFS2 文件系统也将恢复读写。

此时，在 web2 节点通过 cman\_tool 查看集群状态，信息如下：

```
[root@web2 ~]# cman_tool status
```

Version: 6.2.0

Config Version: 40

Cluster Name: mycluster

Cluster Id: 56756

Cluster Member: Yes

Cluster Generation: 3092

Membership state: Cluster-Member

**Nodes: 3**

**Expected votes: 6**

**Quorum device votes: 2**

**Total votes: 5**

**Quorum: 3**

Active subsystems: 9

Flags: Dirty

Ports Bound: 0 177

Node name: web2

Node ID: 1

Multicast addresses: 239.192.221.146

Node addresses: 192.168.12.240

从输出可知，集群节点数变为 3，同时 Quorum 值也有原来的 4 变为 3，Quorum 值是通过



$N/2$  整除加一得到的，其中 N 为所有集群节点的投票数加上表决磁盘投票值。其实也就在这里的 “Total votes” 值。

接着陆续将 Mysql2、Mysql1 异常宕机，宕机完成，在 web2 节点通过 cman\_tool 查看集群状态，信息如下：

```
[root@web2 ~]# cman_tool status
```

Version: 6.2.0

Config Version: 40

Cluster Name: mycluster

Cluster Id: 56756

Cluster Member: Yes

Cluster Generation: 3100

Membership state: Cluster-Member

**Nodes: 1**

**Expected votes: 6**

**Quorum device votes: 2**

**Total votes: 3**

**Quorum: 2**

**Active subsystems: 9**

Flags: Dirty

Ports Bound: 0 177

Node name: web2

Node ID: 1

Multicast addresses: 239.192.221.146

Node addresses: 192.168.12.240

从输出可知，整个集群系统节点数已经变为一个，但是集群系统仍然可用，GFS2 仍然可以正常读写，这些都是 Qdisk 的功能，一个 RHCS 集群中，如果没有表决磁盘的话，仅剩一个节点的集群是不能工作的。

## 二、存储集群测试

GFS2 文件系统是 RHCS 集群中的共享存储，在集群各个节点挂载 GFS2 共享文件系统，然后在任意节点对共享分区进行数据的读写操作，例如：

在节点 web2 上创建一个文件 ixdba：

```
[root@web2 gfs2]# echo "This is GFS2 Files test" >/gfs2/ixdba
```

然后在节点 web1 查看此文件

```
[root@web1 gfs2]# more /gfs2/ixdba
```

```
This is GFS2 Files test
```

可以看到，写操作正常。

接着测试同时读写文件，首先在节点 web1 上编辑文件 ixdba，然后同时在节点 web2 也编辑 ixdba，此时会提示该文件 locked。

到这里为止，RHCS 所有功能点都已经测试完成了，在测试集群功能时，观察日志信息是很有必要的，通过观察日志输出，可以知道 RHCS 更加详细的切换过程，如果切换失败，也会在日志中输出，然后根据错误信息，就可以很容易的定位问题。

目前，RHCS 已经广泛应用在企业的各个领域，与其它 HA 软件相比，RHCS 可能显得比较复杂，管理和维护会比较困难，但是 RHCS 的可靠性和稳定性是毋庸置疑的，随着人们对业务实时性和稳定性需求的不断提升，RHCS 的应用必将越来越广泛。

## 第四篇：维护篇

### 一、启动 RHCS 集群

RHCS 集群的核心进程有 cman 和 rgmanager，要启动集群，依次在集群的每个节点执行如下命令即可：

```
service cman start
```

```
service rgmanager start
```

需要注意的是，执行这两个命令是有先后顺序的，需要首先启动 cman，然后在启动 rgmanager。在集群所有节点成功启动 cman 服务后，然后继续依次在每个节点启动 rgmanager 服务。

### 二、关闭 RHCS 集群

与启动集群服务刚好相反，关闭 RHCS 集群的命令为：

```
service rgmanager stop
```

```
service cman stop
```

首先在集群的每个节点依次关闭 rgmanager 服务，等待所有节点的 rgmanager 服务成功关闭后，再依次关闭每个节点的 cman 服务即可完成整个集群服务的关闭。

有时在关闭 cman 服务时，可能会提示关闭失败，此时可以检查本机的共享存储 GFS2 文件系统是否已经卸载，还可以检查其它节点的 rgmanager 服务是否都已经正常关闭。

### 三、管理应用服务

集群系统启动后，默认是自动启动应用服务的，但是如果某个应用服务没有自动启动，就需要通过手工方式来启动。管理应用服务的命令是 clusvcadm，通过这个命令可以启动、关闭、重启、切换集群中的应用服务。

### 1. 启动某个应用服务

可以通过如下方式启动某个节点的应用服务：

```
clusvcadm -e <Service> -m <Node>
```

其中：

Service：表示集群中创建的应用服务名称。

Node：表示集群节点名称。

例如，要启动节点 web1 上的 webserver 服务，操作如下：

```
[root@web1 ~]# clusvcadm -e webserver -m web1
```

```
Member web1 trying to enable service:webserver...Success
```

```
service:webserver is now running on web1
```

可以通过/var/log/messages 文件查看启动应用服务的详细信息。当 webserver 启动后，与服务相关的集群资源：如虚拟 IP、应用程序服务脚本也随之启动，可以通过如下命令查看集群资源是否已经正常加载：

### 2. 关闭某个应用服务

可以通过如下方式关闭某个节点的应用服务：

```
clusvcadm -s <Service> -m <Node>
```

例如，要关闭节点 Mysql1 上的 mysqlserver 服务，操作如下：

```
[root@Mysql1 ~]# clusvcadm -s mysqlserver -m Mysql1
```

```
Member Mysql1 stopping service:mysqlserver...Success
```

可以通过/var/log/messages 文件查看关闭应用服务的详细信息。当 mysqlserver 关闭后，与服务相关的集群资源：如虚拟 IP、应用程序服务脚本也随之释放。

### 3. 重启某个应用服务

可以通过如下方式重启某个节点的应用服务：

```
clusvcadm -R <Service> -m <Node>
```

例如，要重启节点 web1 上的 webserver 服务，操作如下：

```
[root@web2 ~]# clusvcadm -R webserver -m web1
```

```
Member web1 trying to restart service:webserver...Success
```

这个命令是在 web2 节点上执行的，但是也能成功将 web1 节点上的 webserver 进行重启，由此可知，clusvcadm 命令在集群任意节点执行都是可以的。

#### 4. 切换某个服务

可以通过如下方式将一个应用服务从一个节点切换到另一个节点：

```
clusvcadm -r <Service> -m <Node>
```

例如，要将节点 web1 的服务切换到节点 web2 上，操作如下：

```
[root@web1 ~]# clusvcadm -r webserver -m web2
```

```
Trying to relocate service:webserver to web2...Success
```

```
service:webserver is now running on web2
```

### 四、监控 RHCS 集群状态

通过对 RHCS 的监控，有助于了解集群每个节点的健康状况，并能发现问题，及时解决问题，RHCS 集群提供了丰富的状态查看命令，这里主要介绍下 cman\_tool、clustat、ccs\_tool 的使用方法。

#### 1. cman\_tool 命令

cman\_tool 的参数比较多，但是用法比较简单，基本用法格式为：

```
cman_tool
```

<join|leave|kill|expected|votes|version|wait|status|nodes|services|debug>

[options]

下面列举几个简单的使用例子:

```
[root@web1 ~]# cman_tool nodes -a
```

Node	Sts	Inc	Joined	Name
0	M		0 2010-08-23 01:24:00	/dev/sdb7
1	M	2492	2010-08-23 01:22:43	web2
Addresses: 192.168.12.240				
2	M	2492	2010-08-23 01:22:43	Mysql1
Addresses: 192.168.12.231				
3	M	2492	2010-08-23 01:22:43	Mysql2
Addresses: 192.168.12.232				
4	M	2488	2010-08-23 01:22:43	web1
Addresses: 192.168.12.230				

此命令显示了节点名称, 以及对应的节点 IP 地址和加入集群的时间。

如果要了解更多集群节点信息, 可以通过如下命令:

```
[root@web1 ~]# cman_tool status
```

Version: 6.2.0

Config Version: 35 #集群配置文件版本号

Cluster Name: mycluster #集群名称

Cluster Id: 56756

Cluster Member: Yes

Cluster Generation: 2764

Membership state: Cluster-Member

Nodes: 4       #集群节点数

Expected votes: 6       #期望的投票数

Quorum device votes: 2       #表决磁盘投票值

Total votes: 6       #集群中所有投票值大小

Quorum: 4   #集群法定投票值，低于这个值，集群将停止服务

Active subsystems: 9

Flags: Dirty

Ports Bound: 0 177

Node name: web1

Node ID: 4   #本节点在集群中的 ID 号

Multicast addresses: 239.192.221.146   #集群广播地址

Node addresses: 192.168.12.230   #本节点对应的 IP 地址

## 2. clustat 命令

clustat 命令使用非常简单，详细的使用方法可以通过“clustat -h”获取帮助信息，这里仅仅列举几个例子。

```
[root@web1 ~]#clustat -i 3
```

Cluster Status for mycluster @ Mon Aug 23 18:54:15 2010

Member Status: Quorate

Member Name	ID	Status
-------------	----	--------

-----

-----

-----

```
web2                1      Online, rgmanager
```

```
Mysql1             2      Online, rgmanager
```

```
Mysql2             3      Online, rgmanager
```

```
web1                4      Online, Local, rgmanager
```

```
/dev/sdb7           0      Online, Quorum Disk
```

Service Name	Owner (Last)	State
--------------	--------------	-------

-----

service:mysqlserver	Mysql1	started
---------------------	--------	---------

service:webserver	web1	started
-------------------	------	---------

对输出内容含义如下：

clustat 的“-i”参数可以实时的显示集群系统中每个节点以及服务的运行状态，“-i 3”表示每三秒刷新一次集群状态。

在这个输出中，可以看到每个节点都处于“Online”状态，表明每个节点都运行正常，如果某个节点退出了集群，对应的状态应该是“Offline”，同时还可以看到，集群的两个服务也处于“started”状态，分别运行在 Mysql1 节点和 web1 节点。

另外，通过“ID”一列可以知道集群节点的对应关系，例如，web2 在此集群中对应的就是“Node 1”节点，同理，web1 对应的是“Node 4”节点。了解集群节点顺序有助于对集群日志的解读。

### 3. ccs\_tool 命令

ccs\_tool 主要用来管理集群配置文件 cluster.conf，通过 ccs\_tool 可以在集群中增加/删除节点、增加/删除 fence 设备、更新集群配置文件等操作。

下面是 ccs\_tool 的几个应用实例：



当在一个节点修改完配置文件后，可以执行“`ccs_tool update`”指令将配置文件在所有节点进行更新，例如：

```
[root@web1 cluster]# ccs_tool update /etc/cluster/cluster.conf
```

```
Proposed updated config file does not have greater version number.
```

```
Current config_version :: 35
```

```
Proposed config_version:: 35
```

```
Failed to update config file.
```

`ccs_tool` 是根据 `cluster.conf` 中的“`config_version`”值来决定是否进行更新的，因此在修改完 `cluster.conf` 文件后，一定要将 `cluster.conf` 的 `config_version` 值进行更新，这样执行 `ccs_tool` 时才能更新配置文件。

```
[root@web1 cluster]# ccs_tool update /etc/cluster/cluster.conf
```

```
Config file updated from version 35 to 36
```

```
Update complete.
```

## 五、管理和维护 GFS2 文件系统

GFS2 文件系统提供了很多管理和维护工具，常用的有 `gfs2_fsck`、`gfs2_tool`、`gfs2_jadd`、`gfs2_quota`、`gfs2_convert` 等，这里重点介绍前三个命令的用法。

### 1. `gfs2_fsck` 命令

类似与 `ext3` 文件系统下的 `fsck.ext3` 命令，主要用于检测和修复文件系统错误。其实 GFS2 还有一个 `fsck.gfs2` 命令，此命令与 `gfs2_fsck` 命令完全一致。

`gfs2_fsck` 的用法如下：

```
gfs2_fsck [-afhnpqvVy] <device>
```

下面列举几个使用例子：

```
[root@Mysql1 ~]# gfs2_fsck -y /dev/sdb5
```

```
Initializing fsck
```

```
Validating Resource Group index.
```

```
Level 1 RG check.
```

```
(level 1 passed)
```

```
Starting pass1
```

```
Starting pass1c
```

```
Pass1c complete
```

```
.....
```

```
Pass5 complete
```

```
gfs2_fsck complete
```

## 2. gfs2\_tool 命令

gfs2\_tool 命令参数较多，但使用并不复杂，它主要用来查看、修改 GFS2 文件系统的相关参数信息。

下面列举几个使用例子：

### 1) 查看 GFS2 文件系统挂载信息

```
[root@web1 ~]# gfs2_tool df /gfs2
```

```
/gfs2:
```

```
SB lock proto = "lock_dlm"
```

```
SB lock table = "mycluster:my-gfs2"
```

```
SB ondisk format = 1801
```

SB multihost format = 1900

Block size = 4096

Journals = 4

Resource Groups = 19

Mounted lock proto = "lock\_dlm"

Mounted lock table = "mycluster:my-gfs2"

Mounted host data = "jid=2:id=65539:first=0"

Journal number = 2

Lock module flags = 0

Local flocks = FALSE

Local caching = FALSE

Type		Total Blocks	Used Blocks	Free
Blocks	use%			
-----				
-----		-----	-----	-----
data		1220724	136578	
			1084146	11%
inodes		1084263	117	
			1084146	0%

(2) gfs2\_tool 命令

2) 锁定与解锁 GFS2 文件系统:

```
[root@node1 gfs2]# gfs2_tool freeze /gfs2
```

```
[root@node1 gfs2]# gfs2_tool unfreeze /gfs2
```

GFS2 文件系统被锁定后，无法进行读写操作，直到被解锁。

### 3) 查询 GFS2 可挂载的接点数

```
[root@web1 ~]# gfs2_tool journals /gfs2
```

```
journal2 - 128MB
```

```
journal3 - 128MB
```

```
journal1 - 128MB
```

```
journal0 - 128MB
```

```
4 journal(s) found.
```

这里显示了可挂载节点数为 4，并且每个 journal 的大小为 128M。

### 4) 显示 GFS2 的版本信息：

```
[root@web1 ~]# gfs2_tool version
```

```
gfs2_tool 0.1.62 (built Mar 31 2010 07:34:25)
```

```
Copyright (C) Red Hat, Inc. 2004-2006 All rights reserved
```

### (3) gfs2-jadd 命令

gfs2-jadd 主要用来配置 GFS2 的 Journals 数量和大小，用法非常简单：

```
gfs2_jadd [-cDhJjqV] /path/to/filesystem
```

下面列举几个用例：

设置 Journals 的大小为 64M

```
[root@Mysql1 ~]# gfs2_jadd -J 64M
```

将 GFS2 可同时挂载的节点数目增加到 5 个

```
[root@Mysql1 ~]# gfs2_jadd -j 5 /gfs2
```

另外，另外 `gfs2_quota` 用于 GFS2 文件系统磁盘配额管理，`gfs2_convert` 是一个数据转换应用程序，它可以对 GFS 文件系统的元数据进行更新，把它转换为一个 GFS2 文件系统。

要深入了解它们的使用，请参考帮助信息，这里不在进行讲述。