红帽企业版 Linux 6

集群管理

配置和管理高可用性附加组件

Last Updated: 2017-10-12

红帽企业版 Linux 6 集群管理

配置和管理高可用性附加组件

红帽 工程内容服务
docs-need-a-fix@redhat.com

法律通告

摘要

《配置和管理高可用性附加组件》论述了在红帽企业版 Linux 6 中配置和管理高可用性附加组件。

目录

简介

本文档提供有关安装、配置和管理红帽高可用性附加组件的信息。红帽高可用性附加组件可让您连接到作为集群使用的一组计算机（称之为 *节点*或者 *成员*）。在这个文档中，*集群*或者 *多个集群*指的是运行红帽高可用性附加组件的一组计算机。

本文档的读者应该有丰富的 Red Hat Enterprise Linux 知识，并理解集群、存储和服务器计算的概念。

有关 Red Hat Enterprise Linux 6 的详情请参考以下资源：

- *《Red Hat Enterprise Linux 安装指南》* — 提供有关安装 Red Hat Enterprise Linux 6 的信息。
- *《Red Hat Enterprise Linux 部署指南》* — 提供有关部署、配置和管理 Red Hat Enterprise Linux 6 的信息。

有关 Red Hat Enterprise Linux 6 高可用性附加组件以及相关产品的详情请参考以下资源：

- *《高可用性附加组件概述》* — 提供红帽高可用性附加组件的高级概述。
- *《逻辑卷管理器管理》* — 提供对逻辑卷管理器（LVM）的描述，包括在集群的环境中运行 LVM 的信息。
- *《全局文件系统 2：配置和管理》* — 提供有关安装、配置和维护红帽 GFS2（红帽全局文件系统 2）的信息，GFS2 包含在弹性存储附加组件中。
- *《设备映射器多路径》* — 提供有关使用红帽企业版 Linux 6 设备映射器多路径功能的信息。
- *《负载平衡管理》* — 提供使用负载平衡附加组件配置高性能系统和服务的信息。该组件是一组整合的软件组件，可为在一组真实服务器间平衡 IP 负载提供 Linux 虚拟服务器（LVS）。
- *《发行注记》* — 提供有关红帽产品当前发行本的信息。

高可用性附加组件文档以及其它红帽文档都有 HTML、PDF 和 RPM 版本，您可在红帽企业版 Linux 文档 CD 以及在线文档 http://docs.redhat.com/docs/en-US/index.html 中找到。

1. 反馈

如果您在这本手册中发现任何印刷错误，或者您对本手册有改进意见，我们非常乐于倾听！请在 Bugzilla（http://bugzilla.redhat.com/bugzilla/）中根据组件 **doc-Cluster_Administration** 提交报告。

请确定包含了手册识别符：

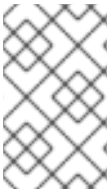
```
Cluster_Administration(EN)-6 (2013-2-15T16:26)
```

通过这个手册识别符，我们可以了解您使用的具体版本。

如果您有针对文档的建议，请尽量具体描述。如果您发现任何错误，请附带章节号以及上下文，以便我们查找。

第 1 章 RED HAT 高可用性附加组件配置和管理概述

Red Hat 高可用性附加组件可让您连接到作为集群使用的一组计算机（称之为 *节点*或者 *成员*）。您可使用 Red Hat 高可用性附加组件满足您的集群需要（例如：为 GFS2 文件系统中的共享文件设置集群，或者设置服务故障切换）。



注意

有关使用高可用附加组件和红帽全局文件系统 2（GFS2）部署和升级红帽企业版 Linux 集群的最佳实践，请参考红帽客户门户网站中的文章《红帽企业版 Linux 集群，高可用性和 GFS 部署最佳实践》，地址为 . https://access.redhat.com/kb/docs/DOC-40821。

本章总结了自红帽企业版 Linux 6 初次发布后在红帽高可用性附加组件中添加的功能和更新文档，并包含 Red Hat 高可用性附加组件配置义管理概述。

1.1. 新的和更改的功能

本小节列出了发布红帽企业版 Linux 6 以后红帽高可用附加组件文档中新的和更改的功能。

1.1.1. 红帽企业版 Linux 6.1 中新的和更改的功能

红帽企业版 Linux 6.1 包括以下文档和功能更新及更改。

- 从红帽企业版 Linux 6.1 开始，红帽高可用性附加组件提供 SNMP 陷阱支持。有关使用红帽高可用性附加组件配置 SNMP 陷阱的详情请参考 第 10 章 *使用红帽高可用性附加组件进行 SNMP 配置*。
- 从红帽企业版 Linux 6.1 发行本开始，红帽高可用性附加组件支持 **ccs** 集群配置命令。有关 **ccs** 命令的详情请参考 第 5 章 *使用 **ccs** 命令配置红帽高可用性附加组件* 和 第 6 章 *使用 **ccs** 管理 Red Hat 高可用性附加组件*。
- 已更新使用 Conga 配置和管理红帽高可用性附加组件软件的文档，包括更新的 Conga 页面和功能支持。
- 从红帽企业版 Linux 6.1 开始，使用 **ricci** 时，需要在首次在任意节点中推广更新的集群配置文件时输入密码。有关 **ricci** 的详情请参考 第 2.13 节 “**ricci** 注意事项”。
- 您现在可为服务指定 *重启-禁用（Restart-Disable）* 失败策略，表示该系统应该在其失败的地方尝试重启，但如果重启服务失败，则会禁用该服务，而不是将其移动到该集群的另一台主机中。这个功能在 第 3.10 节 “在集群中添加集群服务” 和 附录 B, *HA 资源参数* 中有记录。
- 您现在可以将独立子树配置为 non-critical，表示如果该资源失败，只禁用那个资源。有关这个功能的详情请参考 第 3.10 节 “在集群中添加集群服务” 和 第 C.4 节 “故障恢复和独立子树”。
- 这个文档现在包括一个新的章节 第 9 章 *诊断并修正集群中的问题*。

另外对整个文档进行了小的修改和说明。

1.1.2. 红帽企业版 Linux 6.2 中新的和更改的功能

红帽企业版 Linux 6.2 包括以下文档和功能更新及更改。

- 红帽企业版 Linux 现在支持在 active/active 配置中运行集群的 Samba。有关集群的 Samba 的配置详情请参考 第 11 章 *集群 Samba 配置*。

- 虽然所有可在托管 **luci** 的系统中认证的用户都可以登录 **luci**，但从红帽企业版 Linux 6.2 开始，只有运行 **luci** 的系统中的 root 用户可访问 **luci** 的所有组件，除非管理员（root 用户或有管理员权限的用户）为那个用户设定权限。有关为用户设定 **luci** 权限的详情请参考 第 3.3 节 “控制对 luci 的访问”。

- 集群中的节点可以使用 UDP 多播机制彼此沟通。有关配置 UDP 多播的详情请参考 第 2.12 节 “UDP 单播流量”。

- 您现在可以使用 **/etc/sysconfig/luci** 文件配置 **luci** 行为的一些方面。例如：您可以特别配置唯一提供 **luci** 的 IP 地址。有关在唯一 IP 地址提供 **luci** 的配置详情请参考 表 2.2 “在运行 **luci** 的计算机中启用的 IP 端口”。有关 **/etc/sysconfig/luci** 文件的一般信息请参考 第 2.4 节 “使用 **/etc/sysconfig/luci** 配置 **luci**。”。

- **ccs** 命令现在包括 **--lsfenceopts** 选项（输出可用 fence 设备列表）和 **--lsfenceopts** *fence_type* 选项（输出每个可用 fence 类型）。有关这些选项的详情请参考 第 5.6 节 “列出 Fence 设备和 Fence 设备选项”。

- **ccs** 命令现在包括 **--lsserviceopts** 选项（输出您集群现在可用的集群服务列表）和 **--lsserviceopts** *service_type* 选项（输出您可为具体服务类型指定的选项列表）。有关这些选项的详情请参考 第 5.11 节 “列出可用集群服务”。

- 红帽企业版 Linux 6.2 支持 VMware（SOAP 接口）fence 代理。有关 fence 设备参数详情请参考 附录 A, *Fence 设备参数*。

- 红帽企业版 Linux 6.2 在 RHEV 3.0 及之后的版本中支持 RHEV-M REST API fence 代理。有关 fence 设备参数详情请参考 附录 A, *Fence 设备参数*。

- 从红帽企业版 Linux 6.2 开始，您在集群中使用 **ccs** 命令配置虚拟机时，可使用 **--addvm** 选项（而不是 **addservice** 选项）。这样可保证直接在集群配置文件的 **rm** 配置节点中定义 **vm** 资源。有关使用 **ccs** 命令配置虚拟机资源的详情请参考 第 5.12 节 “虚拟机资源”。

- 本文档包括新的附录 附录 D, *集群服务资源检查及故障切换超时*。这个附录描述了 **rgmanager** 如何监控集群资源状态，以及如何修改状态检查间隔。该附录还论述了 **__enforce_timeouts** 服务参数，它可表示操作超时可造成服务失败。

- 本文档包含新的一节 第 2.3.3 节 “配置 iptables 防火墙允许集群组件运行”。在这一节中演示了您可以用来允许多播流量通过 **iptables** 防火墙用于各种集群组件的过滤功能。

另外对整个文档进行了小的修改和说明。

1.1.3. 红帽企业版 Linux 6.3 中新的和更改的功能

红帽企业版 Linux 6.3 包括以下文档和功能更新及更改。

- 红帽企业版 Linux 6.3 支持 **condor** 资源代理。有关 HA 资源参数详情请参考 附录 B, *HA 资源参数*。

- 这个文档现在包括一个新的附录 附录 F, *高可用性 LVM（HA-LVM）*。

- 贯穿本文档的信息明确说明配置更改需要集群重启方可生效。有关这些更改的小结请参考 第 9.1 节 “配置更改不生效”。

- 本文档现在指出如果您与 **luci** 15 分钟内没有互动则会有闲置超时，使您退出程序。有关启动 **luci** 的详情请参考 第 3.2 节 “启动 **luci**”。

- **fence_ipmilan** fence 设备支持特权等级参数。有关 fence 设备参数详情请参考 附录 A, *Fence 设备参数*。
- 这个文档现在包括新的一节 第 2.14 节 “在集群的环境中配置虚拟机”。
- 这个文档现在包括新的一节 第 4.6 节 “备份和恢复 luci 配置”。
- 这个文档现在包括新的一节 第 9.4 节 “集群守护进程崩溃”。
- 本文档在 第 5.14.4 节 “日志”、第 7.7 节 “配置 Debug 选项” 和 第 9.13 节 “需启用发布式锁定管理器（DLM）的 Debug 日志” 中提供设置 debug 选项的信息。
- 从红帽企业版 Linux 6.3 开始，root 用户或者有 **luci** 管理员权限的用户还可以使用 **luci** 界面在该系统中添加用户，如 第 3.3 节 “控制对 luci 的访问” 所示。
- 从红帽企业版 Linux 6.3 开始，**ccs** 命令根据您使用 **-h** 选项所指定节点中 **/usr/share/cluster/cluster.rng** 文件的集群方案验证配置。之前 **ccs** 命令总是使用打包在 **ccs** 命令中的集群方案，即本地系统中的 **/usr/share/ccs/cluster.rng**。有关配置验证的详情请参考 第 5.1.6 节 “配置验证”。
- **cluster.conf** 文件中现在包括描述 附录 A, *Fence 设备参数* 中 fence 设备参数的表格以及描述 附录 B, *HA 资源参数* 中 HA 资源参数的表格，以及那些参数的名称。

另外对整个文档进行了小的修改和说明。

1.1.4. 红帽企业版 Linux 6.4 中新的和更改的功能

红帽企业版 Linux 6.4 包括以下文档和功能更新及更改。

- 红帽企业版 Linux 6.4 发行本提供对 Eaton 网络电源控制器（SNMP 接口）fence 代理、HP 刀片机系统 fence 代理以及 IBM iPDU fence 代理的支持。有关 fence 设备参数的详情请参考 附录 A, *Fence 设备参数*。
- 附录 B, *HA 资源参数* 现提供 NFS 服务器资源代理描述。
- 从红帽企业版 Linux 6.4 开始，root 用户或者有 **luci** 管理员权限的用户还可以使用 **luci** 界面在该系统中删除用户，本文档地址为 第 3.3 节 “控制对 luci 的访问”。
- 附录 B, *HA 资源参数* 提供用于文件系统和 GFS2 HA 资源的新 **nfsrestart** 参数描述。
- 本文档包含新的一节 第 5.1.5 节 “覆盖之前设置的命令”。
- 第 2.3 节 “启用 IP 端口” 现包含用于 **igmp** 的 **iptables** 防火墙过滤信息。
- IPMI LAN fence 代理现支持在 IPMI 设备中配置特权等级的参数，如 附录 A, *Fence 设备参数* 所述。
- 除以太网捆绑模式 1 外，目前还支持集群中的内部节点以捆绑模式 0 和 2 进行沟通。本文档中的故障排除建议您确定只使用现在列出的支持捆绑模式。
- 现在支持使用标记为 VLAN 的网络设备用于集群心跳沟通。故障排除中的建议显示并未从这个文档中删除此项支持。
- 红帽高可用附加组件现在支持冗余环协议配置。有关使用这个功能以及配置 **cluster.conf** 配置文件的信息请参考 第 7.6 节 “配置冗余环协议”。有关使用 **luci** 配置冗余环协议的详情请参考 第 3.5.4 节 “配置冗余环协议”。有关使用 **ccs** 命令配置冗余环协议的详情请参考 第 5.14.5 节 “配

置冗余环协议”。

另外对整个文档进行了小的修改和说明。

1.2. 配置基础

要设置集群，您必须将节点连接到某些集群硬件，并将该节点配置到集群环境中。配置和管理 Red Hat 高可用性附加组件包括以下基本步骤：

1. 设置硬件。请参考 第 1.3 节 “设置硬件”。
2. 安装 Red Hat 高可用性附加组件软件。请参考 第 1.4 节 “安装 Red Hat 高可用性附加组件软件”。
3. 配置 Red Hat 高可用性附加组件软件。请参考 第 1.5 节 “配置 Red Hat 高可用性附加组件软件”。

1.3. 设置硬件

设置硬件包括将集群节点连接到需要运行 Red Hat 高可用性附加组件的其他硬件。硬件的数量和类型根据集群目的和可用性要求有所不同。通常企业级集群需要以下硬件类型（请参考 图 1.1 “Red Hat 高可用性附加组件硬件概述”）。如考虑硬件和其他集群配置问题，请参考 第 2 章 *配置红帽高可用性附加组件前的准备工作*，或者咨询授权 Red Hat 代表。

- 集群节点— 可运行 Red Hat Enterprise Linux 6 软件，且至少有 1GB 内存的计算机。
- 以太网开关或者公用网络集线器 — 为访问该集群的客户端所需。
- 以太网开关或者专用网络集线器 — 为集群节点间通讯以及其他集群硬件，比如网络电源开关和光纤开关所需。
- 网络电源开关 — 建议在企业级集群中使用网络电源开关执行 fencing。
- 光纤开关 — 光纤开关提供对光纤存储的访问。根据存储接口的不同还有其他可用存储选项，例如：iSCSI。可将光纤开关配置为执行 fencing。
- 存储 — 一些集群所需存储类型，具体类型要视集群目的而定。

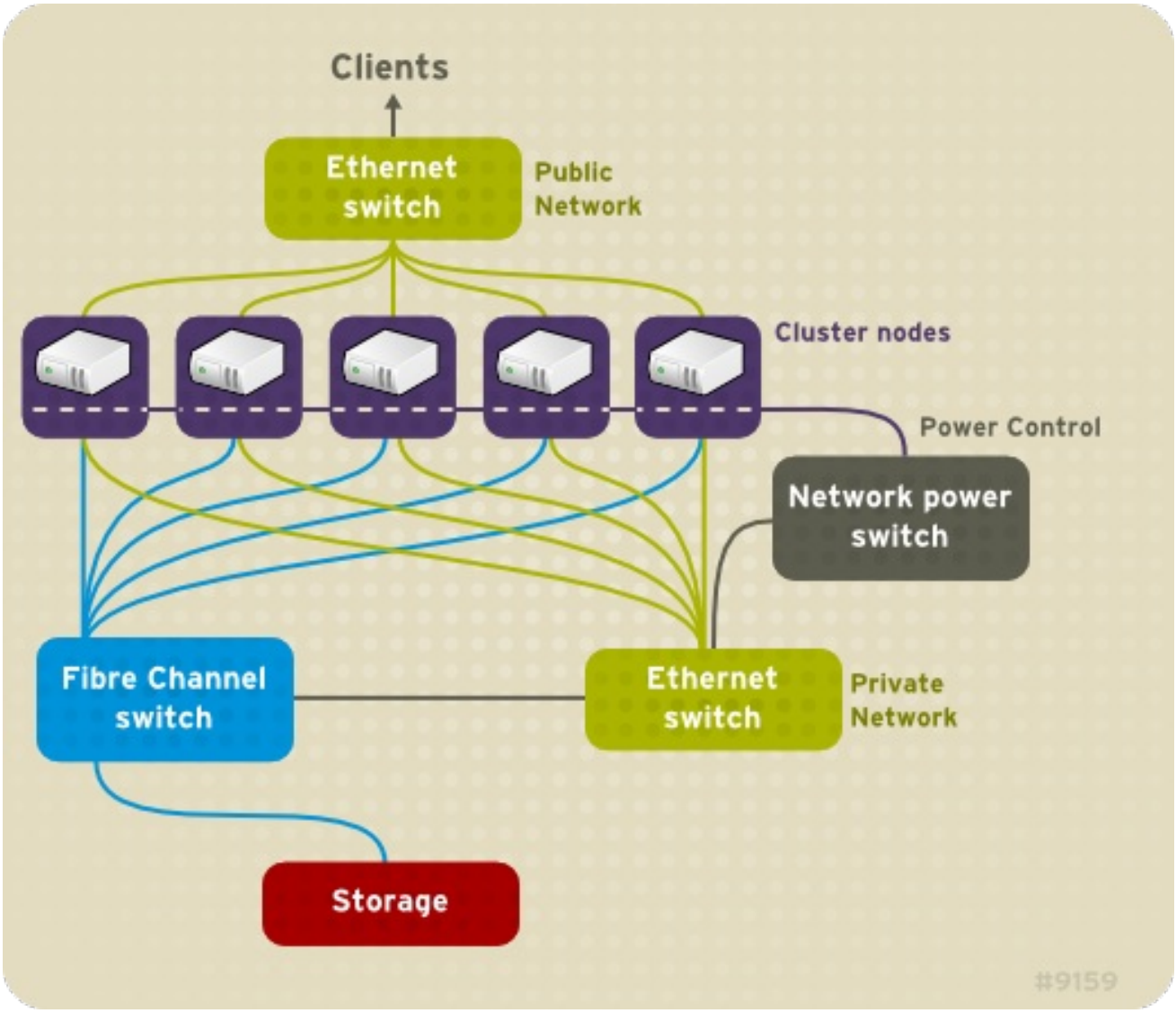


图 **1.1. Red Hat 高可用性附加组件硬件概述**

1.4. 安装 RED HAT 高可用性附加组件软件

安装 Red Hat 高可用性附加组件软件，您必须有该软件的授权。如果您使用 **luci** 配置 GUI，则可以让它安装集群软件。如果您使用其他工具配置该集群，请使用 Red Hat Enterprise Linux 软件锁定并安装该软件。

您可以使用以下 **yum install** 命令安装红帽高可用附加组件软件包：

```
# yum install rgmanager lvm2-cluster gfs2-utils
```

注：只安装 **rgmanager** 将牵动所有所需相依性软件以便在高可用频道中创建 HA 集群。**lvm2-cluster** 和 **gfs2-utils** 软件包是弹性存储频道的一部分，您并不需要它们。

升级 Red Hat 高可用性附加组件软件

可在给定红帽企业版 Linux 主要发行本中不单独提出集群的情况下升级集群软件。这样做需要每次在一台主机中禁用该集群软件，升级该软件并在那台主机中重启该进群软件。

1. 在单一集群节点中关闭所有集群服务。有关在一个节点中停止集群软件的步骤，请参考 第 8.1.2 节 “停止集群软件”。停止 **rgmanager** 前最好手动重新定位集群管理的服务，并使虚拟机脱离该主机。

2. 执行 **yum update** 命令更新已安装软件包。

3. 手动重启集群节点或者集群服务。有关在节点中启动集群服务的步骤请参考 第 8.1.1 节 “启动集群软件”。

1.5. 配置 RED HAT 高可用性附加组件软件

配置 Red Hat 高可用性附加组件软件包括使用配置工具指定集群组件之间的关系。以下集群配置工具可用于 Red Hat 高可用性附加组件：

- **Conga** — 这是一个用于安装、配置和管理 Red Hat 高可用性附加组件的综合用户界面。有关使用 **Conga** 配置和管理高可用性附加组件的详情请参考 第 3 章 *使用 **Conga** 配置红帽高可用性附加组件* 和 第 4 章 *使用 **Conga** 管理 Red Hat 高可用性附加组件*。

- **ccs** — 这个命令配置和管理 Red Hat 高可用性附加组件。有关使用 **ccs** 配置和管理高可用性附加组件的详情请参考 第 5 章 *使用 **ccs** 命令配置红帽高可用性附加组件* 和 第 6 章 *使用 **ccs** 管理 Red Hat 高可用性附加组件*。

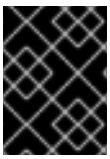
- 命令行工具 — 这是一组配置和管理 Red Hat 高可用性附加组件的命令行工具。有关使用命令行工具配置和管理集群的详情请参考 第 7 章 *使用命令行工具配置红帽高可用附加组件* 和 第 8 章 *使用命令行工具管理红帽高可用性附加组件*。首选命令行工具小结请参考 附录 E, *命令行工具小结*。

注意

在 RHEL 6 中不能使用 **system-config-cluster**。

第 2 章 配置红帽高可用性附加组件前的准备工作

本章论述了安装和配置红帽高可用性附加组件前要执行的任务及注意事项，由以下小节组成：



重要

确定您的红帽高可用性附加组件部署满足您的需要并可获得支持。部署前请联络授权红帽代表确认您的配置。另外请预留充分时间测试失败模式。

2.1. 常规配置注意事项

您可使用各种方法配置红帽高可用性附加组件以满足您的需要。当您进行计划、配置和实施您的部署时，请考虑以下常规注意事项：

支持的集群节点数

红帽高可用性附加组件最多支持的集群节点数为 16。

单点集群

现在只能完全支持单点集群。官方尚不支持在多个物理位置中分布的集群。有关详情以及多点集群的讨论，请联络您的红帽销售或者支持代表。

GFS2

虽然 GFS2 文件系统既可作为独立系统使用，也可作为集群配置的一部分，但红帽不支持将 GFS2 作为单节点文件系统使用。红帽支持很多为单节点优化的高性能单节点文件系统，它们相对集群文件系统来说支出更低。红帽建议您在那些只需要在单一节点挂载文件系统时首选那些系统，而不是

GFS2。红帽将继续为现有客户支持单节点 GFS2 文件系统。

当您将 GFS2 文件系统作为集群文件系统配置时，您必须确定该集群中的所有节点都可访问共享的文件系统。不支持不对称集群配置，在不对称集群中，有些节点可访问该文件系统，而其他节点则不能。这不要求所有节点确实挂载该 GFS2 文件系统。

无单点故障硬件配置

集群可包括一个双控制器 RAID 阵列、多绑定链路、集群成员和存储间的多路径以及冗余无间断供电（UPS）系统以保证没有单点故障造成的应用程序失败或者数据丢失。

另外，可设置一个低消耗集群以提供比无单点故障集群低的可用性。例如：您可以设置一个使用单控制器 RAID 阵列和只使用单以太网链路的集群。

某些低消耗备选方案，比如主机 RAID 控制器、无集群支持的软件 RAID 以及多启动器平行 SCSI 配置与共享集群存储不兼容，或者不适合作为共享集群存储使用。

确保数据完整

要保证数据完整，则每次只能有一个节点可运行集群服务和访问集群服务数据。在集群硬件配置中使用电源开关，就可让一个节点在故障切换过程中，重启节点 HA 服务前为另一个节点提供动力。这样就可防止两个节点同时访问同一数据并破坏数据。强烈建议使用 *Fence 设备*（远程供电、关闭和重启集群节点的硬件或者软件解决方案），以确保在所有失败情况下数据的完整性。

以太网通道绑定

集群仲裁以及节点是否正常运行是由在通过以太网在集群节点间的沟通信息确定的。另外，集群节点使用以太网执行各种重要集群功能（例如：fencing）。使用以太网通道绑定，可将多个以太网接口配置为作为一个接口动作，这样就减小了在集群节点间以及其他集群硬件间典型切换的以太网连接单点故障风险。

从红帽企业版 Linux 6.4 开始支持模块 0、1 和 2。

IPv4 和 IPv6

高可用性附加组件支持 IPv4 和 IPv6 互联网协议。在高可用性附加组件中支持 IPv6 是红帽企业版 Linux 6 的新功能。

2.2. 兼容的硬件

配置红帽高可用性附加组件软件前，请确定您的集群使用合适的硬件（例如：支持 fence 设备、存储设备以及光纤开关等等）。有关大多数当前硬件兼容性信息，请参考硬件配置指南 http://www.redhat.com/cluster_suite/hardware/。

2.3. 启用 IP 端口

部署红帽高可用性附加组件前，您必须在集群以及运行 **luci**（**Conga** 用户界面服务器）的计算机中启用某些 IP 端口。以下小节指出要启用的 IP 端口：

- 第 2.3.1 节 “在集群节点中启用 IP 端口”
- 第 2.3.2 节 “为 **luci** 启用 IP 端口”

下面一节提供红帽高可用性附加组件所需的启用 IP 端口的 **`iptables`** 规则：

- 第 2.3.3 节 “配置 iptables 防火墙允许集群组件运行”

2.3.1. 在集群节点中启用 IP 端口

要允许集群中的节点彼此之间沟通，您必须启用为特定红帽高可用性附加组件分配的 IP 地址。表 2.1 “在红帽高可用性附加组件节点中启用 IP 端口” 列出 IP 端口号，其各自的协议以及分配给那些组件的端口号。在每个集群节点根据 表 2.1 “在红帽高可用性附加组件节点中启用 IP 端口” 启用端口号。您可以使用 **system-config-firewall** 启用 IP 端口。

表 2.1. 在红帽高可用性附加组件节点中启用 IP 端口

IP 端口号	协议	组件
5404, 5405	UDP	**corosync/cman**（集群管理器）
11111	TCP	**ricci**（推广更新的集群信息）
21064	TCP	**dlm**（发布的锁定管理器）
16851	TCP	**modclusterd**

2.3.2. 为 luci 启用 IP 端口

要让客户端计算机与运行 **luci**（**Conga** 用户界面服务器）的计算机沟通，您必须启用分配给 **luci** 的 IP 端口。请在每台运行 **luci** 的计算机中根据 表 2.2 “在运行 **luci** 的计算机中启用的 IP 端口” 启用 IP 端口。

注意

如果集群节点正在运行 **luci**，则应该已经启用了端口 11111。

表 2.2. 在运行 luci 的计算机中启用的 IP 端口

IP 端口号	协议	组件
8084	TCP	**luci**（**Conga** 用户接口服务器）

从使用 **/etc/sysconfig/luci** 工具启用配置的红帽企业版 Linux 6.1 开始，您可以只配置提供 **luci** 的 IP 地址。如果您的服务器基础设施整合多个网络，且您希望只通过内部网络访问 **luci**，则您可以使用这个功能。要做到这一点，请在该文件中取消注释并编辑指定 **host** 的行。例如：要将该文件中的 **host** 设置改为 10.10.10.10，请根据如下操作编辑 **host** 行：

```
host = 10.10.10.10
```

有关 **/etc/sysconfig/luci** 文件的详情请参考 第 2.4 节 “使用 **/etc/sysconfig/luci** 配置 **luci**。”。

2.3.3. 配置 iptables 防火墙允许集群组件运行

以下列出的是红帽企业版 Linux 6（包含高可用性附加组件）所需启用 IP 端口的 iptable 规则示例。请注意这些示例使用 192.168.1.0/24 作为子网，但如果您使用这些规则，就需要使用适当的子网替换 192.168.1.0/24。

请为 **cman**（集群管理器）使用以下过滤。

```
$ iptables -I INPUT -m state --state NEW -m multiport -p udp -s
192.168.1.0/24 -d 192.168.1.0/24 --dports 5404,5405 -j ACCEPT
$ iptables -I INPUT -m addrtype --dst-type MULTICAST -m state --state NEW
-m multiport -p udp -s 192.168.1.0/24 --dports 5404,5405 -j ACCEPT
```

对于 **dlm**（发布的锁定管理程序）：

```
$ iptables -I INPUT -m state --state NEW -p tcp -s 192.168.1.0/24 -d
192.168.1.0/24 --dport 21064 -j ACCEPT
```

为 **ricci**（Conga 远程代理的一部分）：

```
$ iptables -I INPUT -m state --state NEW -p tcp -s 192.168.1.0/24 -d
192.168.1.0/24 --dport 11111 -j ACCEPT
```

对于 **modclusterd**（Conga 远程代理的一部分）：

```
$ iptables -I INPUT -m state --state NEW -p tcp -s 192.168.1.0/24 -d
192.168.1.0/24 --dport 16851 -j ACCEPT
```

对于 **luci**（Conga 用户界面服务器）：

```
$ iptables -I INPUT -m state --state NEW -p tcp -s 192.168.1.0/24 -d
192.168.1.0/24 --dport 16851 -j ACCEPT
```

对于 **igmp**（互联网组管理协议）：

```
$ iptables -I INPUT -p igmp -j ACCEPT
```

执行这些命令后，请运行以下命令保存当前配置以便可在重启后保留这些更改。

```
$ service iptables save ; service iptables restart
```

2.4. 使用 /ETC/SYSCONFIG/LUCI 配置 LUCI。

从红帽企业版 Linux 6.1 开始，您可以使用 **/etc/sysconfig/luci** 文件配置 **luci** 行为的某些方面。您可以在这个文件中更改参数，包括 init 脚本使用的运行环境辅助设置以及服务器配置。另外，您可以编辑这个文件以修改某些应用程序配置参数。在该文件中对如何编辑这个文件来更改配置参数有具体的描述。

为保护预期的格式，您在编辑 **/etc/sysconfig/luci** 文件时不要更改非配置行。另外，您还要小心遵守这个文件要求的语法，特别是在 **INITSCRIPT** 部分，在这部分中不允许等号前后有空格，同时要求您使用引号将包含空格的字符串括起来。

以下示例演示了如何通过编辑 **/etc/sysconfig/luci** 文件更改提供**luci** 的端口。

1. 在 **/etc/sysconfig/luci** 文件中取消注释以下行：

   ```
   #port = 4443
   ```

2. 使用所需端口号替换 4443，该端口号必须大于等于 1024（不是特权端口）。例如：您可以编辑该文件的那一行，将提供 **luci** 的端口设定为 8084。

```
port = 8084
```

3. 重启 **luci** 以便更改生效。

重要

当在 **/etc/sysconfig/luci** 中修改配置参数以重新定义默认值时，请在使用新值替换文档中所用默认值时格外小心。例如：当您修改提供 **luci** 的端口时，请确定您在为 **luci** 启用 IP 端口时指定的是修改后的值，如 第 2.3.2 节 “为 **luci** 启用 IP 端口” 所述。

luci 服务启动时会自动在显示的 URL 中体现修改的端口和主机参数，如 第 3.2 节 “启动 **luci**” 所述。您可以使用这个 URL 访问 **luci**。

有关您可以使用 **/etc/sysconfig/luci** 文件进行配置的参数的完整信息请参考该文件中的描述。

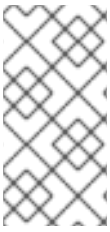
2.5. 将 ACPI 配置为使用整合的 FENCE 设备

如果您的集群使用整合的 fence 设备，则您必须配置 ACPI（高级配置和电源接口）以保证迅速和完全的 fencing。

注意

有关红帽高可用性附加组件支持的整合 fence 设备的最新信息请参考 http://www.redhat.com/cluster_suite/hardware/。

如果将集群节点配置为使用整合的 fence 设备执行 fence 操作，则需要为该节点禁用 ACPI 软关机。禁用 ACPI 软关机可让整合的 fence 设备立即完全关闭节点，而不是尝试彻底关闭（例如：**shutdown -h now**）。否则，如果启用了 ACPI 软关闭，整合的 fence 设备将至少消耗 4 秒时间关闭一个节点（请参考下面的备注）。另外，如果启用 ACPI 软关闭，且在关闭过程中出现节点 panic 或者停滞，则整合的 fence 设备将无法关闭该节点。在那些情况下，fencing 操作将被延迟或者失败。结果是当使用整合的 fence 设备对节点执行 fence 操作并启用 ACPI 软关闭时，集群恢复会很慢并需要管理员介入方可恢复。



注意

Fence 一个节点所需时间取决于所使用的整合 fence 设备。有些整合 fence 设备的功能与长按电源开关一致，因此 fence 设备可在 4-5 秒内关闭该节点。其他整合 fence 设备性能与按一下电源开关一致，要依靠操作系统关闭该节点，因此 fence 设备关闭该节点的时间要大大超过 4-5 秒钟。

要禁用 ACPI 软关闭，请使用 **chkconfig** 管理，并确认在执行 fence 操作后可立即关闭该节点。禁用 ACPI 软关闭的首选方法是使用 **chkconfig** 管理。但如果这个方法不适用于您的集群，您可以使用以下备选方法之一禁用 ACPI 软关闭：

- 将 BIOS 设置改为 "无延迟关闭（instant-off）" 或与之对等的设置以便在没有延迟的情况下关闭该节点

注意

使用 BIOS 禁用 ACPI 软关闭可能不适用于某些计算机。

- 在 **/boot/grub/grub.conf** 文件的内核引导命令行中附加 **acpi=off**

重要

这个方法可完全禁用 ACPI。有些计算机在完全禁用 ACPI 时无法正常引导。*只有*在其他方法对您的集群都无效时才使用这个方法。

以下小节提供禁用 ACPI 软关闭的首选和备用方法步骤：

- 第 2.5.1 节 “使用 **chkconfig** 管理禁用 ACPI 软关闭” — 首选方法
- 第 2.5.2 节 “使用 BIOS 禁用 ACPI 软关闭” — 第一备选方法
- 第 2.5.3 节 “在 **grub.conf** 文件中完全禁用 ACPI。” — 第二备选方法

2.5.1. 使用 chkconfig 管理禁用 ACPI 软关闭

您可以使用 **chkconfig** 管理，通过在 **chkconfig** 管理中删除 ACPI 守护进程（**acpid**），或者关闭 **acpid** 禁用 ACPI 软关闭。

注意

这是禁用 ACPI 软关闭的首选方法。

使用 **chkconfig** 管理在每个集群节点中禁用 ACPI 软关闭，如下：

1. 运行以下命令之一：
   - **chkconfig --del acpid** — 这个命令会从 **chkconfig** 管理中删除 **acpid**。

     — 或者 —

   - **chkconfig --level 2345 acpid off** — 这个命令关闭 **acpid**。
2. 重启该节点。
3. 当配置并运行集群时，请确认在执行 fence 时立即关闭该节点。

注意

您可使用 **fence_node** 命令或者 **Conga** fence 该节点。

2.5.2. 使用 BIOS 禁用 ACPI 软关闭

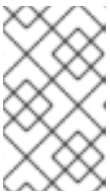
禁用 ACPI 软关闭的首选方法是 **chkconfig** 管理（第 2.5.1 节 “使用 **chkconfig** 管理禁用 ACPI 软关闭”）。但如果首选方法对您的集群无效，请按照本节中的步骤执行。

注意

使用 BIOS 禁用 ACPI 软关闭可能不适用于某些计算机。

您可以通过配置每个集群节点中的 BIOS 禁用 ACPI 软关闭，如下：

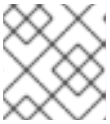
1. 重启该节点并启动 **BIOS CMOS Setup Utility** 程序。
2. 浏览**「Power」**菜单（或者对等的电源管理菜单）
3. 在**「Power」**菜单中将**「Soft-Off by PWR-BTTN」**功能（或者对等的功能）改为**「Instant-off」**（或者使用电源按钮无延迟关闭节点的对等设置）。在 例 2.1 “**BIOS CMOS Setup Utility**：将**「Soft-Off by PWR-BTTN」**设定为**「Instant-Off」**。” 中演示将**「Power」**菜单中的**「ACPI Function」**设定为**「Enabled」**，并将**「Soft-Off by PWR-BTTN」**设定为**「Instant-Off」**。

   注意

   与**「ACPI Function」**、**「Soft-Off by PWR-BTTN」**和**「Instant-Off」**对等的菜单在不同计算机中会有所不同。但这个步骤的目的是配置 BIOS 以便计算机可使用电源开关无延迟地关闭计算机。

4. 退出 **BIOS CMOS Setup Utility** 程序，保存 BIOS 配置。
5. 当配置并运行集群时，请确认在执行 fence 时立即关闭该节点。

   注意

   您可使用 **fence_node** 命令或者 **Conga** fence 该节点。

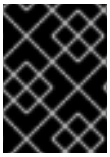
例 2.1. BIOS CMOS Setup Utility：将「Soft-Off by PWR-BTTN」设定为「Instant-Off」。

```
+---------------------------------------------|-------------------+
|    ACPI Function             [Enabled]      |    Item Help      |
|    ACPI Suspend Type         [S1(POS)]      |-------------------|
|  x Run VGABIOS if S3 Resume   Auto          |   Menu Level   *  |
|    Suspend Mode              [Disabled]     |                   |
|    HDD Power Down            [Disabled]     |                   |
|    Soft-Off by PWR-BTTN      [Instant-Off   |                   |
|    CPU THRM-Throttling       [50.0%]        |                   |
|    Wake-Up by PCI card       [Enabled]      |                   |
|    Power On by Ring          [Enabled]      |                   |
|    Wake Up On LAN            [Enabled]      |                   |
|  x USB KB Wake-Up From S3     Disabled      |                   |
|    Resume by Alarm           [Disabled]     |                   |
|  x  Date(of Month) Alarm       0            |                   |
|  x  Time(hh:mm:ss) Alarm       0 :  0 :     |                   |
|    POWER ON Function         [BUTTON ONLY   |                   |
|  x KB Power ON Password       Enter         |                   |
|  x Hot Key Power ON           Ctrl-F1       |                   |
|                                             |                   |
|                                             |                   |
+---------------------------------------------|-------------------+
```

这个示例演示了如何将「**ACPI Function**」设定为「**Enabled**」；将「**Soft-Off by PWR-BTTN**」设定为「**Instant-Off**」。

2.5.3. 在 grub.conf 文件中完全禁用 ACPI。

禁用 ACPI 软关闭的首选方法是使用 **chkconfig** 管理（第 2.5.1 节 “使用 **chkconfig** 管理禁用 ACPI 软关闭”）。如果首选的方法不适用于您的集群，您可以使用 BIOS 电源管理（第 2.5.2 节 “使用 BIOS 禁用 ACPI 软关闭”）禁用 ACPI 软关闭。如果这两种方法都不适用于您的集群，您可以在 **grub.conf** 文件的内核引导命令行中附加 **acpi=off**，这样就可以完全禁用 ACPI。



重要

这个方法可完全禁用 ACPI。有些计算机在完全禁用 ACPI 时无法正常引导。*只有*在其他方法对您的集群都无效时才使用这个方法。

您可以通过编辑每个集群节点的 **grub.conf** 文件完全禁用 ACPI，如下：

1. 使用文本编辑器中打开 **/boot/grub/grub.conf**。
2. 在 **/boot/grub/grub.conf** 的内核引导命令行中附加 **acpi=off**（请参考 例 2.2 “附加了 **acpi=off** 的内核引导命令行”）。
3. 重启该节点。
4. 当配置并运行集群时，请确认在执行 fence 时立即关闭该节点。

注意

您可使用 **fence_node** 命令或者 **Conga** fence 该节点。

例 2.2. 附加了 acpi=off 的内核引导命令行

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this
file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/mapper/vg_doc01-lv_root
#          initrd /initrd-[generic-]version.img
#boot=/dev/hda
default=0
timeout=5
serial --unit=0 --speed=115200
terminal --timeout=5 serial console
title Red Hat Enterprise Linux Server (2.6.32-193.el6.x86_64)
        root (hd0,0)
        kernel /vmlinuz-2.6.32-193.el6.x86_64 ro
root=/dev/mapper/vg_doc01-lv_root console=ttyS0,115200n8 acpi=off
        initrd /initramrs-2.6.32-131.0.15.el6.x86_64.img
```

> 在这个示例中，在内核引导命令行中附加了 **acpi=off** — 即以 "kernel /vmlinuz-2.6.32-193.el6.x86_64.img" 开始的行。

2.6. 配置 HA 服务注意事项

您可以通过配置 HA（高可用性）服务创建集群以满足您对高可用性的要求。红帽高可用性附加组件中的 HA 服务管理关键组件 **rgmanager** 可为现成的应用程序部署冷故障切换。在红帽高可用性附加组件中，使用其他集群资源配置的应用程序可组成一个 HA 服务，该服务可在集群节点间进行故障切换而不中断集群客户端。如果一个集群节点失败，或者集群系统管理员在集群节点间移动该服务时（例如：计划的集群节点断电），则会发生 HA 服务故障切换。

要创建 HA 服务，您必须在集群配置文件中配置它。HA 服务由集群 *资源*组成。集群资源是在集群配置文件中创建和管理的构建块 — 例如：IP 地址、应用程序初始化脚本或者红帽 GFS2 共享分区。

HA 服务每次只能在一个集群节点中运行以保持数据完整性。您可以在故障切换域中指定故障切换优先权。指定故障切换优先权包括在故障切换域中为每个节点分配优先权等级。如果您没有指定故障切换优先权，则 HA 服务可故障切换到其故障切换域中的任何节点。另外，您还可以指定是否将 HA 服务严格限制在与其故障切换域关联的节点中。（当与非限制故障切换域关联时，HA 服务可在故障切换域所有成员都不可用的事件中在任意节点中启动。）

图 2.1 “网页服务器集群服务示例” 演示名为 "content-webserver" 的网页服务器的 HA 服务示例。它在集群节点 B 中运行，且其故障切换域由节点 A、B 和 D 组成。另外，使用故障切换优先权将故障切换域配置为节点 D 的优先权高于节点 A，并限制为只切换到故障切换域中的节点。HA 服务由以下集群资源组成：

- IP 地址资源 — IP 地址 10.10.10.201。
- 名为 "httpd-content" 的应用程序资源 — 网页服务器应用程序初始化脚本 **/etc/init.d/httpd**（指定 **httpd**）。
- 文件系统资源 — 名为 "gfs2-content-webserver" 的 Red Hat GFS2。

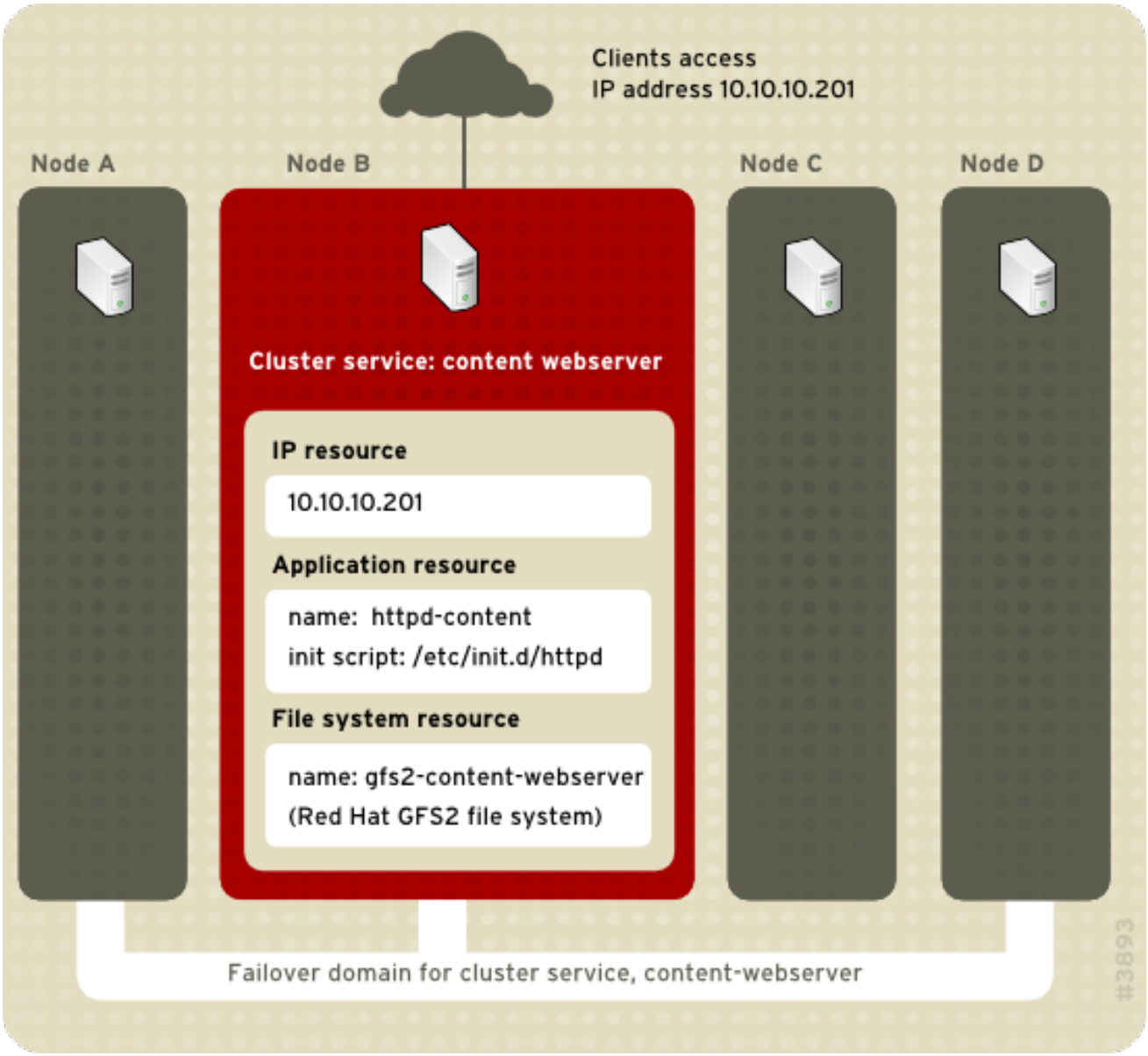


图 **2.1. 网页服务器集群服务示例**

客户端使用 IP 地址 10.10.10.201 访问 HA 服务，启用与网页服务器程序 httpd-content 间的互动。httpd-content 程序使用 gfs2-content-webserver 文件系统。如果节点 B 失败，则 content-webserver HA 服务会切换到节点 D。如果节点 D 不可用或者也失败了，则该服务会切换到节点 A。故障切换会尽量减小集群客户端的服务中断。例如：在 HTTP 服务中，可能会丢失某些状态信息（比如会话数据）。在另一个集群节点中可使用与故障切换前相同的 IP 地址访问 HA 服务。

注意

有关 HA 服务以及故障切换域的详情请参考*《高可用附加组件概述》*。有关配置故障切换域的详情请参考 第 3 章 *使用 **Conga** 配置红帽高可用性附加组件*（使用 **Conga**）或者 第 7 章 *使用命令行工具配置红帽高可用附加组件*（使用命令行工具）。

HA 服务是一组在统一实体中配置的集群资源，可为客户端提供特定的服务。HA 服务在集群配置文件 **/etc/cluster/cluster.conf**（在每个集群节点中）中以资源树的形式出现。在集群配置文件中，每个资源树都使用一个 XML 代表，可指定每个资源及其属性和在资源树中与其它资源的关系（上级、下级和平级关系）。

注意

因为 HA 服务由分为层次树的资源组成，所以服务有时也指的是*资源树*或者*资源组*。这两个词组与 *HA 服务*是同义词。

在每个资源树的顶端是一个特殊的资源类型 — *服务资源*。其他资源类型构成服务的其他部分，决定服务的特点。配置 HA 服务包括创建服务资源、生成下级集群资源以及将其组成统一实体形成该服务的分级限制。

配置 HA 服务时两个主要考虑的问题：

- 创建服务所需的资源类型
- 资源间的上级、下级和同级关系

资源类型以及资源的结构由您所要配置服务的类型决定。

集群资源位于 附录 B, *HA 资源参数*。有关资源间上级、下级和同级关系的描述，请参考 附录 C, *HA 资源行为*。

2.7. 配置验证

可在启动和重新载入配置时，根据集群方案 **/usr/share/cluster/cluster.rng** 自动验证集群配置。您还可以使用 **ccs_config_validate** 命令在随时验证集群配置。有关使用 **ccs** 命令时的配置验证详情请参考 第 5.1.6 节 “配置验证”。

您可在 **/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html** 中查看注释的可用方案（例如：**/usr/share/doc/cman-3.0.12/cluster_conf.html**）。

配置验证可检查以下基本错误：

- XML 验证 — 查看该配置文件是否为有效 XML 文件。
- 配置选项 — 查看选项（XML 元素和属性）是否有效。
- 选项值 — 查看选项是否包含有效数据（受限制的）

以下示例为演示有效性检查的有效配置和无效配置：

- 有效配置 — 例 2.3 “**cluster.conf** 示例配置：有效文件”
- 无效 XML — 例 2.4 “**cluster.conf** 示例配置：无效 XML”
- 无效选项 — 例 2.5 “**cluster.conf** 示例配置：无效选项”
- 无效选项值 — 例 2.6 “**cluster.conf** 示例配置：无效选项值”

例 2.3. cluster.conf 示例配置：有效文件

```
<cluster name="mycluster" config_version="1">
  <logging debug="off"/>
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
```

```
          </fence>
      </clusternode>
      <clusternode name="node-02.example.com" nodeid="2">
          <fence>
          </fence>
      </clusternode>
      <clusternode name="node-03.example.com" nodeid="3">
          <fence>
          </fence>
      </clusternode>
    </clusternodes>
    <fencedevices>
    </fencedevices>
    <rm>
    </rm>
</cluster>
```

例 2.4. cluster.conf 示例配置：无效 XML

```
<cluster name="mycluster" config_version="1">
  <logging debug="off"/>
    <clusternodes>
      <clusternode name="node-01.example.com" nodeid="1">
          <fence>
          </fence>
      </clusternode>
      <clusternode name="node-02.example.com" nodeid="2">
          <fence>
          </fence>
      </clusternode>
      <clusternode name="node-03.example.com" nodeid="3">
          <fence>
          </fence>
      </clusternode>
    </clusternodes>
    <fencedevices>
    </fencedevices>
    <rm>
    </rm>
<cluster>         <----------------INVALID
```

在这个示例中，配置的最后一行（在此注释为 "INVALID"）缺少一个斜杠 — 应该是 **</cluster>** 而不是 **<cluster>**。

例 2.5. cluster.conf 示例配置：无效选项

```
<cluster name="mycluster" config_version="1">
  <loging debug="off"/>         <----------------INVALID
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
   </fencedevices>
   <rm>
   </rm>
<cluster>
```

在这个示例中配置的第二行（在此注释为 "INVALID"）包含无效 XML 元素 — 应该是 **logging** 而不是 **loging**。

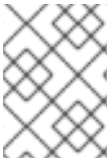
例 2.6. cluster.conf 示例配置：无效选项值

```
<cluster name="mycluster" config_version="1">
  <loging debug="off"/>
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="-1">  <--------
INVALID
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
   </fencedevices>
   <rm>
   </rm>
<cluster>
```

在这个示例中，配置的第四行（在此注释为 "INVALID"）包含 XML 属性的无效值，即 **node-01.example.com** 的 **clusternode** 行中的 **nodeid**。该值应该是一个正数（"1"）而不是负数（"-1"）。**nodeid** 属性值必须是一个正数。

2.8. NETWORKMANAGER 注意事项

不支持在集群节点中使用 **NetworkManager**。如果您已经在集群节点中安装了 **NetworkManager**，您应该删除或者禁用该程序。

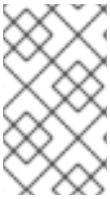


注意

如果 **NetworkManager** 正在运行，或者将其配置为与 **chkconfig** 命令一同运行，则不会启动 **cman** 服务。

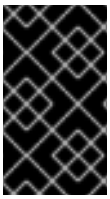
2.9. 使用仲裁磁盘的注意事项

仲裁磁盘是使用磁盘的仲裁守护进程 **qdiskd**，它可提供补充的试探法（heuristics）以确定节点是否正常运作。使用这些试探法，您可以确定在网络分区事件中对节点操作十分重要的因素。例如：在一个按 3:1 分割的有四个节点的集群中，最初三个节点自动"获胜"，因为三对一的占优。在那些情况下，只有一个节点被 fence。但使用 **qdiskd**，您可以设定试探法以便允许一个节点因访问重要资源获胜（例如：关键网络路径）。如果您的集群需要额外的方法确定节点工作正常，那么您应该将 **qdiskd** 配置为满足那些要求。



注意

不需要配置 **qdiskd**，除非您对节点正常工作有特殊的要求。例如："all-but-one" 配置。在 all-but-one 配置中会将 **qdiskd** 配置为提供足够的仲裁投票以便在只有一个节点工作时维护仲裁。



重要

总之，用于您部署的试探法以及其他 **qdiskd** 参数要根据所需环境和特殊要求而定。要了解试探法和其他 **qdiskd** 参数，请参考 qdisk(5) man page。如果您在了解和使用 **qdiskd** 方面需要帮助，请联络授权 Red Hat 支持代表。

如果您需要使用 **qdiskd**，请考虑以下注意事项：

集群节点投票

使用仲裁磁盘时，每个集群节点都必须有一票。

CMAN 成员超时值

CMAN 成员超时值（即 CMAN 认为节点已死，并不再是成员前该节点不响应的时间）应该至少为 **qdiskd** 成员超时值的两倍。理由是仲裁守护进程必须自己探测失败的节点，且比 CMAN 要花更多的时间。CMAN 成员超时默认值为 10 秒。其他具体位置条件可能会影响 CMAN 和 **qdiskd** 成员超时值之间的关系。要在调整 CMAN 成员超时值方面获得帮助，请联络授权 Red Hat 支持代表。

Fencing

要在使用 **qdiskd** 时保证可靠的 fencing，请使用电源 fencing。虽然其他类型的 fencing 在没有配置 **qdiskd** 的集群中可靠，但它们并不适用于配置了 **qdiskd** 的集群。

最多节点数

配置了 **qdiskd** 的集群最多可支持 16 个节点。这个限制的原因是因为扩展性的需要，增加节点计数会增加共享仲裁磁盘设备中的同步 I/O 竞争。

仲裁磁盘设备

仲裁磁盘设备应该是集群中所有节点可同时读取/写入的共享块设备。该块设备最小为 10MB。**qdiskd** 可使用的共享块设备示例有多端口 SCSI RAID 阵列、光纤 RAID SAN 或者配置为 RAID 的 iSCSI 目标。您可以使用群集仲裁磁盘工具 **mkqdisk** 创建仲裁磁盘设备。有关使用该工具的详情请参考 mkqdisk(8) man page。

注意

不建议使用 JBOD 作为仲裁磁盘。JBOD 无法提供可靠的性能，因此可能无法保证节点迅速写入仲裁磁盘。如果某个节点无法迅速写入仲裁磁盘设备，则会错误地将该节点从集群中驱除。

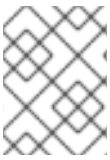
2.10. 红帽高可用性附加组件及 SELINUX

红帽企业版 Linux 6 的高可用性附加组件在将 SELinux 策略类型设定为 **targeted** 时支持 SELinux 的 **enforcing** 状态。

有关 SELinux 的详情请参考红帽企业版 Linux 6 *《部署指南》*。

2.11. 多播地址

集群中的节点使用多播地址彼此沟通。因此必须将红帽高可用附加组件中的每个网络切换以及关联的联网设备配置为启用多播地址并支持 IGMP（互联网组管理协议）。请确定红帽高可用附加组件中的每个网络切换以及关联的联网设备都支持多播地址和 IGMP。如果是这样，请确定启用多播地址和 IGMP。没有多播地址以及 IGMP，则不是所有节点都可成为集群的一部分，从而导致集群失败：在这些环境中使用 UDP 单播，如 第 2.12 节 “UDP 单播流量” 所述。



注意

配置网络开关以及相关联网设备的步骤在每个产品中都有所不同。请参考正确的零售商文档或者其他有关配置网络开关以及相关联网设备启用多播地址和 IGMP 的信息。

2.12. UDP 单播流量

从红帽企业版 Linux 6.2 开始，集群中的节点可以使用 UDP 单播传输机制进行沟通。但建议您在集群网络中使用 IP 多播。UDP 单播是 IP 多播不可用时的备用方法。

您可以在 **cluster.conf** 配置文件中设置 **cman transport="udpu"** 参数，将红帽高可用附加组件配置为使用 UDP 单播。您还可以在 **Conga** 用户界面的**「网络配置」**页面中指定单播，如 第 3.5.3 节 “网络配置” 所述。

2.13. RICCI 注意事项

红帽企业版 Linux 6 中使用 **ricci** 替换 **ccsd**。因此必需在每个集群节点中都运行 **ricci** 方可推广更新的集群配置，您可以使用 **cman_tool version -r**、 **ccs** 命令或者 **luci** 用户界面服务器推广。 可以使用 **service ricci start** 启动 **ricci**，也可以在引导时使用 **chkconfig** 启动它。有关为 **ricci** 启

用 IP 端口的详情请参考 第 2.3.1 节 “在集群节点中启用 IP 端口”。

从红帽企业版 Linux 6.1 开始，您在任意节点中使用 **ricci** 推广更新的集群配置时要求输入密码。您在系统中安装 **ricci** 后，请使用 **passwd ricci** 命令为用户 **ricci** 将 **ricci** 密码设定为 root。

2.14. 在集群的环境中配置虚拟机

当您使用虚拟机资源配置集群时，应使用 **rgmanager** 工具启动和停止虚拟机。使用 **virsh** 启动该机器可导致该虚拟机在多个位置中运行，并造成虚拟机中的数据崩溃。

要减少管理员在集群的环境中使用集群和非集群根据意外“重复启动”虚拟机的机会，您可以将您的系统配置为在非默认位置保存虚拟机配置文件。在非默认位置保存虚拟机配置文件可让意外使用 **virsh** 启动虚拟机变得更困难，因为 **virsh** 很难识别该配置文件。

虚拟机配置文件的非默认位置可以是任意位置。使用 NFS 共享或共享的 GFS2 文件系统的优点是管理员不需要在集群成员间同步该配置文件。但也可以使用本地目录，只要该管理员保证可在集群范围内同步其内容即可。

在集群配置中，虚拟机可使用虚拟机资源的 **path** 属性参考非默认位置。注：**path** 属性是一个目录或一组使用帽号（‘:’）分开的目录，不是到具体文件的路径。

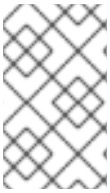
警告

应在所有运行 **rgmanager** 的节点中禁用 **libvirt-guests** 服务。如果某台虚拟机自动启动或恢复，则可导致该虚拟机在多个位置运行，从而造成虚拟机中的数据崩溃。

有关虚拟机资源属性的详情请参考 表 B.24 “虚拟机”。

第 3 章 使用 CONGA 配置红帽高可用性附加组件

本章论述如何使用 **Conga** 配置红帽高可用性附加组件。有关使用 **Conga** 管理运行的集群的详情请参考 第 4 章 *使用 **Conga** 管理 Red Hat 高可用性附加组件*。



注意

Conga 是您可用来管理红帽高可用性附加组件的图形用户界面。请注意，要有效使用这个界面，您需要对一些基础概念有明确和完整的理解。不建议您在用户界面中摸索可用功能了解集群配置，因为这样可使系统在组件失败时不足以保证所有服务运行。

本章由以下小节组成：

3.1. 配置任务

使用 **Conga** 配置红帽高可用性附加组件包括以下步骤：

1. 配置并运行 **Conga** 配置用户界面 — **luci** 服务器。请参考 第 3.2 节 “启动 **luci**”。
2. 创建集群。请参考 第 3.4 节 “创建集群”。
3. 配置全局集群属性。请参考 第 3.5 节 “全局集群属性”。
4. 配置 fence 设备。请参考 第 3.6 节 “配置 Fence 设备”。
5. 为集群成员配置 fencing。请参考 第 3.7 节 “为集群成员配置 Fencing”。
6. 创建故障切换域。请参考 第 3.8 节 “配置故障切换域”。
7. 创建资源。请参考 第 3.9 节 “配置全局集群资源”。
8. 创建集群服务。请参考 第 3.10 节 “在集群中添加集群服务”。

3.2. 启动 LUCI

注意

使用 **luci** 配置集群要求在集群节点中安装并运行 **ricci**，如 第 2.13 节 “**ricci** 注意事项” 所述。在该小节中指出，使用 **ricci** 需要一个密码，您在创建集群时需要为每个集群节点输入该密码，如 第 3.4 节 “创建集群” 所述。

在启动 **luci** 前，请确定您集群节点中的 IP 端口允许任意与 **luci** 沟通的节点中的 **luci** 服务器到端口 11111 的连接。有关在集群节点中启用 IP 端口的详情请参考 第 2.3.1 节 “在集群节点中启用 IP 端口”。

要使用 **Conga** 管理红帽高可用性附加组件，请安装并运行 **luci**，如下：

1. 选择托管 **luci** 的主机，并在那台计算机中安装 **luci** 软件。例如：

```
# yum install luci
```

注意

通常是服务器架或者托管 **luci** 的数据中心的一台计算机中，但集群计算机也可托管 **luci**。

2. 使用 **service luci start** 启动 **luci**。例如：

```
# service luci start
Starting luci: generating https SSL certificates...  done
                                                           [  OK
]

Please, point your web browser to https://nano-01:8084 to access
luci
```

注意

从红帽企业版 Linux 6.1 开始，您可以使用 **/etc/sysconfig/luci** 文件配置 **luci** 行为的某些方面，包括端口和主机参数，如 第 2.4 节 “使用 **/etc/sysconfig/luci** 配置 **luci**。” 所示。修改的端口和主机参数在启动 **luci** 服务时自动在显示的 URL 中体现。

3. 在网页浏览器的地址栏中输入 **cman** 服务器的 URL，并点击 **Go**（或者相当的按钮）。**luci** 服务器的 URL 语法为 **https://*luci_server_hostname*:*luci_server_port***。*luci_server_port* 的默认值为 **8084**。

   您首次访问 **luci** 时，网页浏览器会根据显示的自我签名 SSL 证书（**luci** 服务器的证书）给出具体提示。确认一个或者多个对话框后，您的网页显示器会显示 **luci** 登录页面。

4. 虽然所有可以在托管 **luci** 的系统中认证的用户都可以登录 **luci**，但从红帽企业版 Linux 6.2 开始，只有运行 **luci** 的系统中的 root 可以访问所有 **luci** 组件，除非管理员（root 用户或者有管理员权限的用户）为那个用户设置权限。有关为用户设置 **luci** 权限的详情请参考 第 3.3 节 “控制对 luci 的访问”。

   登录 **luci** 后，**luci** 会显示「**Homebase**」页面，如 图 3.1 “luci Homebase 页面” 所示。

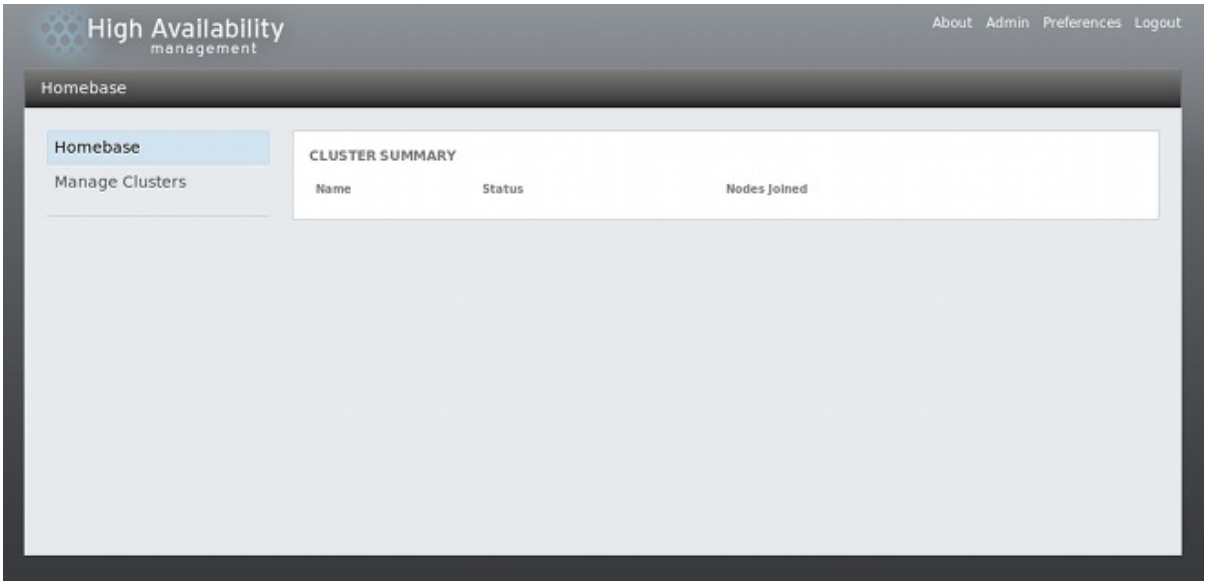


图 3.1. luci Homebase 页面

注意

如果 15 分钟后没有互动，则 **luci** 会处于闲置超时而让您退出。

3.3. 控制对 LUCI 的访问

从红帽企业版 Linux 6 的最初发行本开始就在 **「用户和权限」**页面中添加了以下功能。

- 从红帽企业版 Linux 6.2 开始，root 用户或者运行 **luci** 的系统中有 **luci** 管理员权限的用户都可以通过为系统中的独立用户设定权限访问各种 **luci** 组件。
- 从红帽企业版 Linux 6.3 开始，root 用户或者有 **luci** 管理员权限的用户还可以使用 **luci** 界面在该系统中添加用户。
- 从红帽企业版 Linux 6.4 开始，root 用户或者有 **luci** 管理员权限的用户还可以使用 **luci** 界面从该系统中删除用户。

要添加用户、删除用户或者设定用户权限，请作为 **root** 用户或者之前赋予管理员权限的用户登录 **luci**，并点击 **luci** 页面右上角的**「管理」**选项。此时会出现**「用户和权限」**页面，该页面中显示现有用户。

要删除用户，请选择要删除的用户并点击 **删除所选**。

要添加用户，请点击 **添加用户**，并输入要添加用户的名称。

要为用户设定或者更改权限，请从 **「用户权限」**页面的下拉菜单中选择该用户。这样就可允许您设置以下权限：

点击 Luci 管理员

为该用户授予和 root 用户相同的权限，即在所有集群中有所有权限，并可以在其他所有用户中设置或删除权限，root 用户除外，它的权限是不能被限制的。

点击 可创建集群

允许用户创建新集群，如 第 3.4 节 “创建集群” 所述。

可导入现有集群

允许用户在 **luci** 界面中添加现有集群，如 第 4.1 节 “在 luci 界面中添加现有集群” 所述。

每个在 **luci** 中生成或导入其中的集群，您都可以为指示的用户设置以下权限：

可查看该集群

允许该用户查看指定的集群。

可更改集群配置

允许该用户为指定集群修改配置，但不能添加和删除集群节点。

可启用、禁用、重新定位和迁移服务组

允许用户管理高可用服务，如 第 4.5 节 “管理高可用性服务” 所述。

可停止、启动和重启集群节点

允许用户管理集群的独立节点，如 第 4.3 节 “管理集群节点” 所述。

可添加和删除节点

允许用户在集群中添加和删除节点，如 第 3.4 节 “创建集群” 所述。

可从 Luci 中删除这个集群

允许用户从 **luci** 界面中删除集群，如 第 4.4 节 “启动、停止、刷新和删除集群” 所述。

点击 **提交** 以便权限生效，或者点击 **重置** 返回原始值。

3.4. 创建集群

使用 **luci** 创建集群包括命名集群、在集群中添加集群节点、为每个节点输入 **ricci** 密码并提交创建集群请求。如果节点信息和密码正确，则 **Conga** 会自动在集群节点中安装软件（如果当前没有安装适当的软件包）并启动集群。按如下步骤创建集群：

1. 在 **luci** 「**Homebase**」 页面左侧菜单中点击 「**管理集群**」 。此时会出现 「**集群**」 页面，如 图 3.2 “luci 集群管理页面” 所示。

图 3.2. luci 集群管理页面

2. 点击**「创建」**后出现**「创建集群页面」**，如 图 3.3 “创建 luci 集群对话框” 所示。

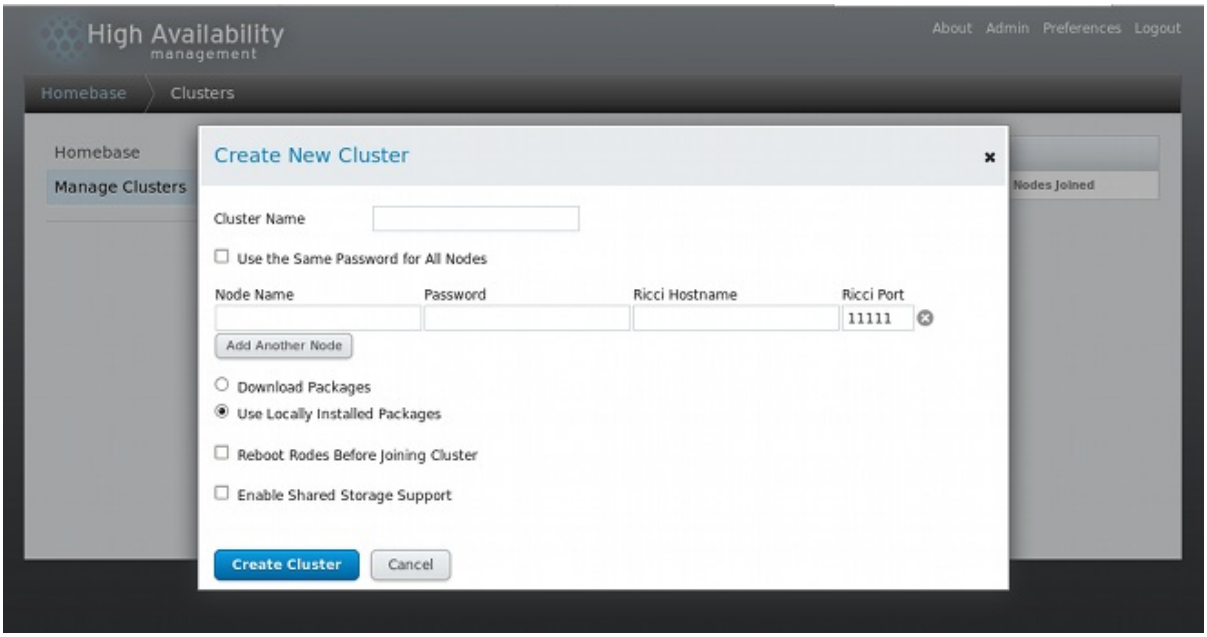


图 3.3. 创建 luci 集群对话框

3. 请根据需要在**「创建新集群」**页面中输入以下参数：

   - 在**「集群名称」**文本框中输入集群名称。集群名称不能超过 15 个字符。
   - 如果集群中的每个节点都有同样的 **ricci** 密码，您可以选择**「在所有节点中使用相同的密码」**，这样就可在您添加的节点中自动填写**「密码」**字段。
   - 在**「节点名称」**栏中输入集群中节点的名称，并在**「密码」**栏中为该节点输入 **ricci** 密码。
   - 如果为您的系统配置了专门用于集群流量的专门的私有网络，则最好将 **luci** 配置为使用与集群节点名称解析拨通的地址与 **ricci** 进行沟通。您可以在**「Ricci 主机名」**中输入该地址达到此目的。

- 如果您要在 **ricci** 代理中使用不同的端口，而不是默认的 11111 端口，您可以更改那个参数。
- 点击**「添加另一个节点」**并输入节点名称，同时为集群的每个附加节点输入 **ricci** 密码。
- 如果您不想要在创建集群时升级已经在节点中安装的集群软件软件包，请选择**「使用本地安装的软件包」**选项。如果您要升级所有集群软件软件包，请选择**「下载软件包」**选项。

注意

如果缺少任意基本集群组件（**cman**、**rgmanager**、**modcluster** 及其所有相依性软件包），无论是选择**「使用本地安装的软件包」**，还是**「下载软件包」**选项，都会安装它们。如果没有安装它们，则创建节点会失败。

- 需要时选择**「加入集群前重启节点」**。
- 如果需要集群的存储，则请选择**「启动共享存储支持」**。这样做将下载支持集群存储的软件包，并启用集群的 LVM。您应该只能在可访问弹性存储附加组件或者可扩展文件系统附加组件时选择这个选项。

4. 点击 **创建集群**。点击 **创建集群** 后会有以下动作：
   1. 如果您选择**「下载软件包」**，则会在节点中下载集群软件包。
   2. 在节点中安装集群软件（或者确认安装了正确的软件包）。
   3. 在集群的每个节点中更新并传推广群配置文件。
   4. 加入该集群的添加的节点

   显示的信息表示正在创建该集群。当集群准备好后，该显示会演示新创建集群的状态，如 图 3.4 “集群节点显示” 所示。请注意：如果没有在任何节点中运行 **ricci**，则该集群创建会失败。

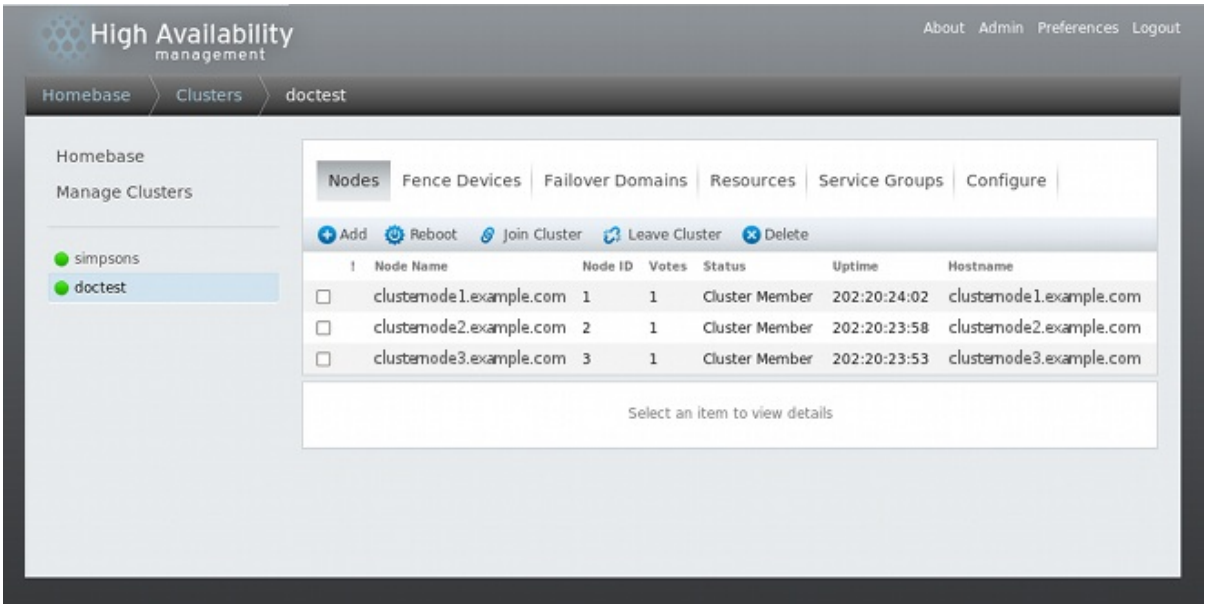
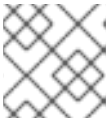


图 3.4. 集群节点显示

5. 点击 **创建集群** 按钮创建集群后，您仍可以通过点击集群节点显示页面上部菜单中的**「添加」**或者**「删除」**功能从集群中添加或者删除节点。除非您要删除整个集群，否则必须在删除节点前停止它们。有关从目前操作中的现有集群中删除节点的详情请参考 第 4.3.4 节 “删除集群中的成

员”。



注意

从集群中删除集群节点是一个破坏性操作，不能撤销。

3.5. 全局集群属性

选择要配置的集群后，会出现该集群的具体页面。该页面提供配置集群范围内属性的界面。您可以点击顶部的**「配置」**链接配置集群范围内的属性。此时会出现有多个标签的页面，这些标签为：**「常规」**、**「Fence 守护进程」**、**「网络」**、**「冗余环」**、**「QDisk」**和**「日志」**。请按照本小节中的步骤配置那些标签中的参数。如果您不需要配置标签中的参数，则请跳过有关那个标签的一节。

3.5.1. 配置常规属性

点击**「常规」**标签显示**「常规属性」**页面，该页面提供修改配置版本的界面。

- **「集群名称」**文本框显示集群名称，它不接受更改集群名称。更改集群名称的唯一方法是创建有新名称的集群配置。
- 默认在创建集群时将**「配置版本」**值设定为 **1**，并在每次修改集群配置后自动增加该值。但如果您需要将其设定为其它值，您可以在**「配置版本」**文本框中指定该值。

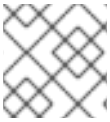
如果您更改了**「配置版本」**值，请点击 **应用** 按钮以便此更改生效。

3.5.2. 配置 Fence 守护进程属性

点击**「Fence 守护进程」**标签显示**「Fence 守护进程属性」**页面，该页面提供配置**「失败后延迟」**和**「加入后延迟」**的界面。您为这些参数配置的值是集群的常规 fencing 属性。要为集群的节点配置具体 fence 设备，请使用集群显示中的**「Fence 设备」**菜单项，如 第 3.6 节 “配置 Fence 设备” 所示。

- **「失败后延迟」**参数为在节点失败之后，执行节点（fence 域中的成员）fencing 前，fence 守护进程（**fenced**）要等待的秒数。**「失败后延迟」**的默认值为 **0**。对不同的集群和网络性能需要可修改该值。
- **「后加入延迟（Post Join Delay）」**参数是该节点加入 fence 守护进程（**fenced**）后，该守护进程 fence 该节点前要等待的秒数。**「后加入延迟」**默认值为 **6**。**「后加入延迟」**一般在 20-30 秒之间，可根据集群和网络性能而有所不同。

输入所需值并点击 **应用** 以便更改生效。



注意

有关**「加入后延迟」**和**「失败后延迟」**的详情请参考 fenced(8) man page。

3.5.3. 网络配置

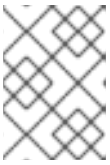
点击**「网络」**标签显示**「网络配置」**页面，该页面提供配置网络传输类型的界面。

您可以使用这个标签选择以下选项之一：

- **「UDP 多播并让集群选择多播地址」**

这是默认设置。选择这个选项，红帽高可用性附加组件软件就会根据集群 ID 创建一个多播地址。它可生成地址的后 16 字节，并根据 IP 协议（IPv4 或者 IPv6）将其附加到地址迁移部分的后面。

- 对于 IPv4 — 该地址格式为 239.192 加上红帽高可用性附加组件软件生成的后 16 字节。
- 对于 IPv6 — 该地址格式为 FF15:: 加上红帽高可用性附加组件软件生成的后 16 字节。



注意

集群 ID 是 **cman** 为每个集群生成的唯一识别符。要查看集群 ID，请在集群节点中运行 **cman_tool status** 命令。

- **「UDP 多播并手动指定多播地址」**

  如果您需要使用具体的多播地址，请选择这个选项并在**「多播地址」**文本框中输入多播地址。

  如果您指定一个多播地址，您应该使用 **cman** 采用的 239.192.x.x 序列（IPv6 使用 FF15::）。否则，使用该范围外的多播地址可能导致无法预计的结果。例如：使用 224.0.0.x（“网络中的所有主机”）可能无法正常路由，有些硬件甚至根本无法路由。

  如果您指定或修改多播地址，则必须重启该集群以便其生效。有关 **Conga** 启动和停止集群的详情请参考 第 4.4 节 “启动、停止、刷新和删除集群”。

  注意

  如果您指定了多播地址，请确定您检查了集群数据包所要经过的路由器。有些路由器需要较长的时间获得地址，这样会严重影响集群性能。

- **「UDP 多播（UDPU）」**

  从红帽企业版 Linux 6.2 开始，集群中的节点可以使用 UDP 单播传输机制进行沟通。但建议您在集群网络中使用 IP 多播。UDP 单播是 IP 多播不可用时的备用方法。不建议在 GFS2 部署中使用 UDP 单播。

点击 **应用**。修改传输类型时，需要重启集群以便更改生效。

3.5.4. 配置冗余环协议

从红帽企业版 Linux 6.4 开始，红帽高可用附加组件支持冗余环协议配置。当使用冗余环协议时，您需要考虑以下事项，如 第 7.6 节 “配置冗余环协议” 所述。

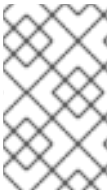
点击**「冗余环」**标签显示**「冗余环协议配置」**页面。该页面显示目前为该集群配置的所有节点。如果您要将某个系统配置为使用冗余环协议，则必须为第二个环在每个节点中指定**「备用名称」**。

「冗余环协议配置」页面还可让您为第二个环指定**「备用环多播地址」**、**「备用环 CMAN 端口」**和**「备用环多播数据包 TTL」**。

如果您为第二个环指定多播地址，要么使用备用多播地址，要么备用端口必须与第一个环的多播地址不同。如果您要指定备用端口，则第一个环和第二个环的端口号之差必须大于 2，因为该系统本身使用端口和端口-1 执行操作。如果您没有指定备用多播地址，该系统会自动为第二个环使用不同的多播地址。

3.5.5. 仲裁磁盘配置

点击「**仲裁磁盘**」标签显示「**仲裁磁盘配置**」页面，该页面可在您需要使用仲裁磁盘时提供配置仲裁磁盘参数的界面。



注意

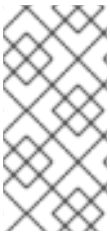
仲裁磁盘参数和试探法要根据具体环境和特殊要求而定。要了解如何使用仲裁磁盘以及试探法，请参考 qdisk(5) man page。如果您在理解和使用仲裁磁盘方面需要帮助，请联络授权的红帽支持代表。

默认启用「**不使用仲裁磁盘**」参数。如果您需要使用仲裁磁盘，请点击「**使用仲裁磁盘**」，输入仲裁磁盘参数，点击「**应用**」并重启该集群以便更改生效。

表 3.1 “仲裁磁盘参数” 描述了仲裁磁盘参数。

表 3.1. 仲裁磁盘参数

参数	描述
「**指定物理设备：根据设备标签**」	指定 **mkqdisk** 程序生成的仲裁磁盘标签。如果使用该字段，则仲裁守护进程会读取 **/proc/partitions** 文件，并在每个找到的块设备中检查仲裁磁盘签名，与指定的标签进行对比。这在节点间使用不同仲裁设备名称时很有用。
「**试探法**」	「**到程序的路径**」 — 用来决定这个试探是否可用的程序。它是 **/bin/sh -c** 可执行的任意程序。返回值为 0 表示成功；其他则表示失败。这是必填项。 「**间隔**」 — 调用试探法的频率（以秒为单位）。每次试探间的默认间隔为 2 秒。 「**分数**」 — 试探法的加权。请小心确定试探法分数。每个试探的默认分数为 1。 「**TKO**」 — 宣布这个试探法不可用前连续失败的次数。
「**最小总分数**」	视节点为“活跃”所需的最小分数。如果忽略或者将其设定为 0，则使用默认功能 **floor((*n*+1)/2)**，其中 *n* 为试探法分数之和。「**最小总分数**」值必须永远小于试探法分数之和，否则将无法使用仲裁磁盘。



注意

点击「**仲裁磁盘配置**」标签中的「**应用**」按钮，将更改推广到每个集群节点的集群配置文件中（**/etc/cluster/cluster.conf**）。但如果要让仲裁磁盘操作，或要对仲裁磁盘参数的修改生效，您就必须重启该集群（请参考 第 4.4 节 “启动、停止、刷新和删除集群”），以保证您在每个节点中都重启 **qdiskd** 守护进程。

3.5.6. 日志配置

点击「**日志**」标签显示「**日志配置**」页面，该页面提供配置日志设置的界面。

您可以为全局日志配置进行以下设置：

- 点击「**记录 debugging 信息**」可启用在日志文件中记录 debugging 信息。

- 点击**「在 syslog 中记录信息」**可启用在 **syslog** 中记录信息的功能。您可以选择**「syslog 信息工具」**和**「syslog 信息优先权」**设置，**「syslog 信息优先权」**设置表示会将所选级别以及更高级别中的信息发送到 **syslog**。

- 点击**「在日志文件中记录信息」**可启用在日志文件中记录信息。您可以指定**「日志文件路径」**名称。**「日志文件信息优先权」**设置表示会将所选级别以及更高级别中的信息写入日志文件。

您可以选择**「日志配置」**页的底部**「具体守护进程日志覆盖」**中列出的一个守护进程覆盖全局日志设置。选择守护进程后，您可以检查是否要为该具体守护进程记录 debugging 信息。您还可以为那个守护进程指定 **syslog** 和日志文件设置。

为日志配置点击 **应用** 以便更改生效。

3.6. 配置 FENCE 设备

配置 fence 设备包括为集群创建、更新和删除 fence 设备。您可在集群中为节点配置 fencing 前必须在集群中配置 fence 设备。

创建 fence 设备包括选择 fence 设备类型以及为那个 fence 设备输入参数（例如：name、IP address、login 和 password）。更新 fence 设备包括选择现有 fence 设备并为那个 fence 设备更改参数。删除 fence 设备包括选择现有 fence 设备并删除它。

本小节提供以下任务的步骤：

- 创建 fence 设备 — 请参考 第 3.6.1 节 “创建 Fence 设备”。创建并命名 fence 设备后，您可以为集群中的每个节点配置 fence 设备，如 第 3.7 节 “为集群成员配置 Fencing” 所述。

- 更新 fence 设备 — 请参考 第 3.6.2 节 “修改 Fence 设备”。

- 删除 fence 设备 — 请参考 第 3.6.3 节 “删除 Fence 设备”。

在具体集群页面中，您可以点击集群显示顶端的**「Fence 设备」**为那个集群配置 fence 设备。这样做可为集群显示 fence 设备，并显示 fence 设备配置菜单项：**「添加」**和**「删除」**。这是以下小节中所有描述步骤的起点。

注意

如果是刚开始集群配置，则还没有创建 fence 设备，因此也没有显示任何 fence 设备。

图 3.5 “luci fence 设备配置页面” 演示在创建 fence 设备前的 fence 设备配置页面。

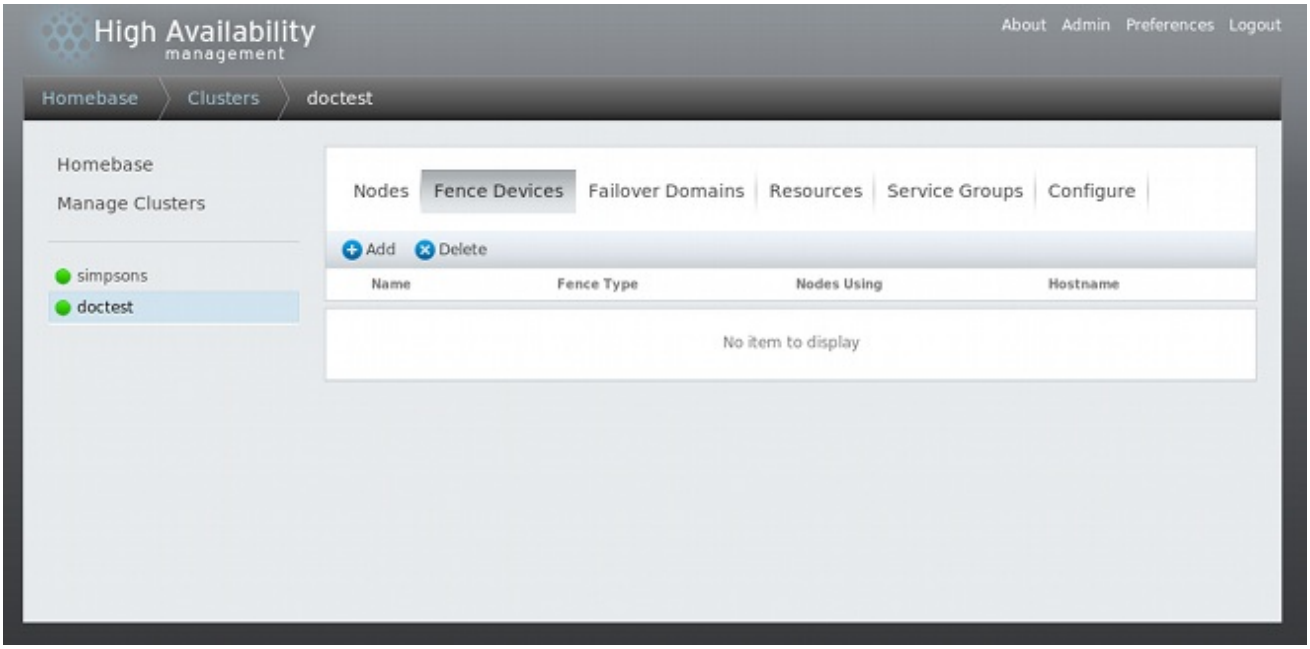


图 **3.5. luci fence 设备配置页面**

3.6.1. 创建 Fence 设备

要创建 fence 设备请按照以下步骤执行：

1. 在「**Fence 设备**」配置页面中，点击「**添加**」。点击「**添加**」显示「**添加 Fence 设备（事务）**」对话框。在这个对话框中选择要配置的 fence 设备类型。

2. 在「**添加 Fence 设备（事务）**」对话框中根据 fence 设备类型指定信息。有关 fence 设备参数详情请参考 附录 A, *Fence 设备参数*。在有些情况下您需要为该 fence 设备指定额外的节点具体参数，如 第 3.7 节 “为集群成员配置 Fencing” 所述。

3. 点击 **提交**。

添加 fence 设备后，它会出现在「**Fence 设备**」配置页面中。

3.6.2. 修改 Fence 设备

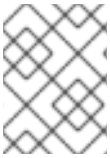
要修改 fence 设备，请按照以下步骤执行：

1. 在「**Fence 设备**」配置页面中点击要修改的 fence 设备名称。此时会出现那个 fence 设备的对话框，该对话框中应该有为该设备配置的值。

2. 要修改 fence 设备，修改显示的参数。有关详情请参考 附录 A, *Fence 设备参数*。

3. 点击 **应用** 并等待更新配置。

3.6.3. 删除 Fence 设备



注意

无法删除使用中的 fence 设备。要删除某个节点目前正在使用的 fence 设备，请首先为使用该设备的所有节点更新节点 fence 配置，然后删除该设备。

要删除 fence 设备，请按照以下步骤执行：

1. 在「**Fence 设备**」配置页面中选择 fence 设备左侧的复选框选择要删除的设备。
2. 点击 **删除** 并等待配置更新。此时会出现一条信息说明已经删除了该设备。

当更新配置后，显示中不再会出现删除的 fence 设备。

3.7. 为集群成员配置 FENCING

您完成创建集群和创建集群 fence 设备的初始步骤后，需要为集群节点配置 fencing。要在创建新集群后为节点配置 fencing 并为其配置 fence 设备，请按照本小节中的步骤执行。请注意：您必须为集群中的每个节点配置 fencing。

以下小节中提供了为节点配置单一 fence 设备、使用备份 fence 设备配置节点以及使用冗余电源配置节点的步骤：

- 第 3.7.1 节 “为节点配置单一 Fence 设备”
- 第 3.7.2 节 “配置备份 Fence 设备”
- 第 3.7.3 节 “配置使用冗余电源的节点”

3.7.1. 为节点配置单一 Fence 设备

使用以下步骤配置有单一 fence 设备的节点。

1. 在具体集群页面中，您可以点击集群显示顶部的「**节点**」，为集群中的节点配置 fencing。这样做会显示组成集群的节点。当您点击 luci 「**Homebase**」页面左侧菜单中的「**管理集群**」项下的集群名称时会出现这个默认页面。
2. 点击节点名称。点击节点链接会出现一个演示如何配置该节点的页面。

   在具体节点页面中显示所有目前在该节点中运行的服务，同时还显示该节点所在故障切换域。您可以点击其名称修改现有故障切换域。有关配置故障切换域详情请参考 第 3.8 节 “配置故障切换域”。
3. 请在具体节点页面的「**Fence 设备**」项下点击 **添加 Fence 方法**。此时会显示 **在节点中添加 Fence 方法** 对话框。
4. 请输入为这个节点配置的 fencing 方法的「**方法名**」。这可以是红帽高可用性附加组件使用的任意名称。这与该设备的 DNS 名称不同。
5. 点击 **提交**。此时会显示具体节点页面，该页面中显示您刚刚在「**Fence 设备**」中添加的方法。
6. 点击 fence 事务下的 **添加 Fence 事务** 标签为这个方法配置 fence 事务。此时会出现「**添加 Fence 设备（事务）**」下拉菜单，您可从中选择您之前配置的 fence 设备，如 第 3.6.1 节 “创建 Fence 设备” 所述。
7. 为这个方法选择 fence 设备。如果这个 fence 设备需要您配置具体节点参数，则会显示要配置的参数。有关 fencing 参数详情请参考 附录 A, *Fence 设备参数*。

注意

对于非电源 fence 方法（即 SAN/存储 fencing），会在具体节点参数显示中默认选择**「取消 fencing（Unfencing）」**。这可保证在重启该节点前不会重新启用被 fence 的节点对存储的访问。有关 unfencing 节点的详情请参考 **fence_node**(8) man page。

8. 点击 **提交**。此时会返回显示 fence 方法和 fence 事务的具体节点页面。

3.7.2. 配置备份 Fence 设备

您可以为一个节点定义多种 fencing 方法。如果使用第一种方法对节点执行 fence 失败，系统会尝试使用第二种方法，随后是您配置的附加方法。

使用以下步骤为节点配置备份 fence 设备。

1. 使用 第 3.7.1 节 “为节点配置单一 Fence 设备” 所述步骤配置节点的主 fencing 方法。
2. 在您定义的主要方法下面点击 **添加 Fence 方法**。
3. 请您为这个节点配置的备份 fence 方法命名并点击 **提交**。此时会出现具体节点，该页面中显示您刚刚在主 fence 方法下添加的方法。
4. 点击 **添加 Fence 事务** 按钮为这个方法配置 fence 事务。此时会出现下拉菜单，您可从中选择您之前配置的 fence 设备，如 第 3.6.1 节 “创建 Fence 设备” 所述。
5. 为这个方法选择 fence 设备。如果这个 fence 设备需要您配置具体节点参数，则会显示要配置的参数。有关 fencing 参数详情请参考 附录 A, *Fence 设备参数*。
6. 点击 **提交**。此时会返回显示 fence 方法和 fence 事务的具体节点页面。

您可以继续根据需要添加 fencing 方法。您可以点击**「上移」**和**「下移」**重新安排这个节点使用的 fencing 方法顺序。

3.7.3. 配置使用冗余电源的节点

如果将集群配置为在节点中使用冗余电源，您必须确定配置了 fencing，这样就可在需要在对节点执行 fence 时将其完全关闭。如果您将每个电源配置为使用独立 fence 的方法，则会分别对每个电源执行 fence 操作。第二个电源可在第一个电源完全被 fence 后允许系统继续运行。要将系统配置为使用双电源，则必须配置您的 fence 设备，以便关闭两个电源时可完全关闭系统。当将系统配置为使用 **Conga** 时要求您在单一 fencing 方法中配置两个事务。

要为有双电源的节点配置 fencing，请按照本小节中的步骤执行。

1. 在您为有冗余电源的节点中配置保护前，您必须将每个电源开关配置为集群的保护设备。有关配置 fence 设备的详情请参考 第 3.6 节 “配置 Fence 设备”。
2. 在具体集群页面中点击集群显示顶端的**「节点」**，此时会显示组成该集群的节点。这也是您点击 **luci「Homebase」**页面左侧菜单中集群名称下的**「管理集群」**时出现的默认页面。
3. 点击节点名称。点击节点链接会出现一个演示如何配置该节点的页面。
4. 在具体节点页面中，请点击 **添加 Fence 方法**。
5. 请输入您为这个节点配置的 fencing 方法名称。

6. 点击 **提交**。此时会显示具体节点页面，该页面中显示您刚刚在「**Fence 设备**」中添加的方法。

7. 点击 **添加 Fence 事务** 将第一个电源供应配置为这个方法的 fence 事务。此时会显示一个下拉菜单，您可从中选择您之前配置的电源 fencing 设备，如 第 3.6.1 节 “创建 Fence 设备” 所示。

8. 为这个方法选择电源 fence 设备之一，并为这个设备输入适当的参数。

9. 点击 **提交**。此时会返回显示 fence 方法和 fence 事务的具体节点页面。

10. 在您为第一个电源 fencing 设备配置的同一 fence 方法中点击 **添加 Fence 事务**。此时会出现一个下拉菜单，您可从中选择您之前配置的第二个电源 fencing 设备，如 第 3.6.1 节 “创建 Fence 设备” 所述。

11. 为这个方法选择第二电源 fence 设备，并为这个设备输入适当的参数。

12. 点击 **提交**。此时会返回具体节点页面，该页面中包括 fence 方法、显示的 fence 事务以及每个系统关闭和打开电源的顺序。如 图 3.6 “双电源 Fencing 配置” 所述。

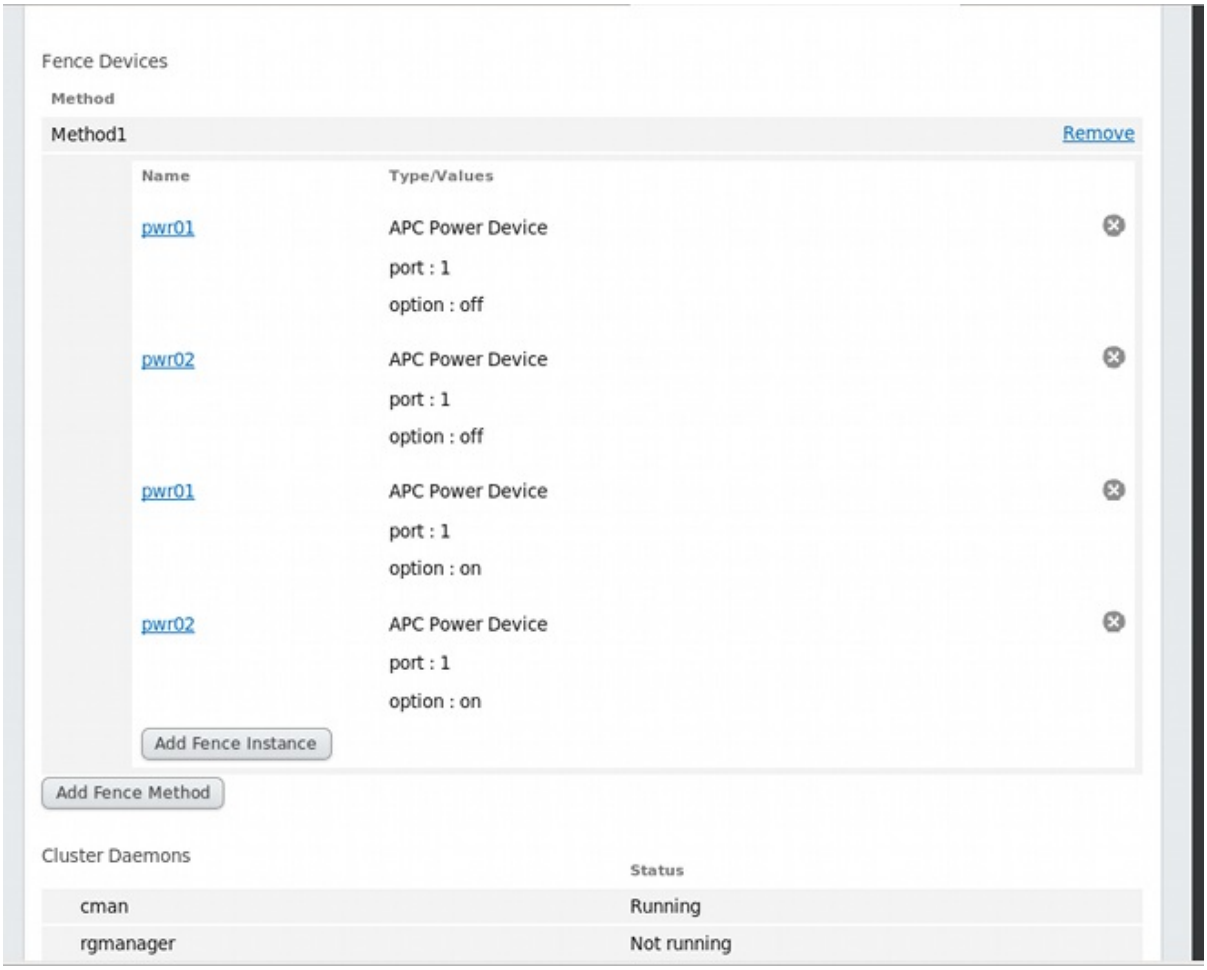


图 3.6. 双电源 Fencing 配置

3.8. 配置故障切换域

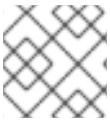
故障切换域是一个命名的集群节点子集，它可在节点失败事件中运行集群服务。故障切换域有以下特征：

- 无限制 — 允许您为在子集指定首选成员子集，但分配给这个域名的集群服务可在任意可用成员中运行。

- 限制 — 允许您限制可运行具体集群服务的成员。如果在限制故障切换域中没有可用成员，则无法启动集群服务（手动或者使用集群软件均不可行）。
- 无序 — 当将一个集群服务分配给一个无序故障切换域时，则可从可用故障切换域成员中随机选择运行集群服务的成员，没有优先顺序。
- 有序的 — 可让您在故障切换域的成员间指定顺序。该列表顶端的成员是首选成员，接下来是列表中的第二个成员，依此类推。
- 故障恢复 — 允许您指定在故障切换域中的服务是否应该恢复到节点失败前最初运行的节点。配置这个特性在作为有序故障切换域一部分节点重复失败的环境中很有帮助。在那种情况下，如果某个节点是故障切换域中的首选节点，在可能在首选节点和其它节点间重复切换和恢复某个服务，从而不会对性能产生严重影响。

注意

故障恢复特性只适用于配置了有序故障切换的集群。



注意

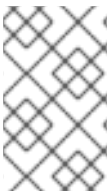
更改故障切换域配置对目前运行中的服务无效。

注意

操作*不需要*的故障切换域。

默认情况下故障切换域为无限制和无序的。

在由几个成员组成的集群中，使用限制故障切换域可最大程度降低设置集群以便运行集群服务的工作（比如 **httpd**），它要求您在运行该集群服务的所有成员中进行完全一致的配置。您不需要将整个集群设置为运行该集群服务，只要设置与该集群服务关联的限制故障切换域中的成员即可。



注意

要配置首选成员，您可以创建只有一个集群成员的无限制故障切换域。这样做就让集群服务主要在那个集群成员（首选成员）中运行，但允许将该集群服务故障切换到任意其它成员中。

以下小节描述了如何添加、修改和删除故障切换域：

- 第 3.8.1 节 “添加故障切换域”
- 第 3.8.2 节 “修改故障切换域”
- 第 3.8.3 节 “删除故障切换域”

3.8.1. 添加故障切换域

要添加故障切换域，请按照本小节中的步骤执行。

1. 在具体集群页面中，您可以点击集群显示顶部的**「故障切换域」**为那个集群配置故障切换域。此时会显示为这个集群配置的故障切换域。

2. 点击**「添加」**。点击**「添加」**时会显示**「在集群中添加故障切换域」**对话框，如 图 3.7 “luci 故障切换域配置对话框” 所示。

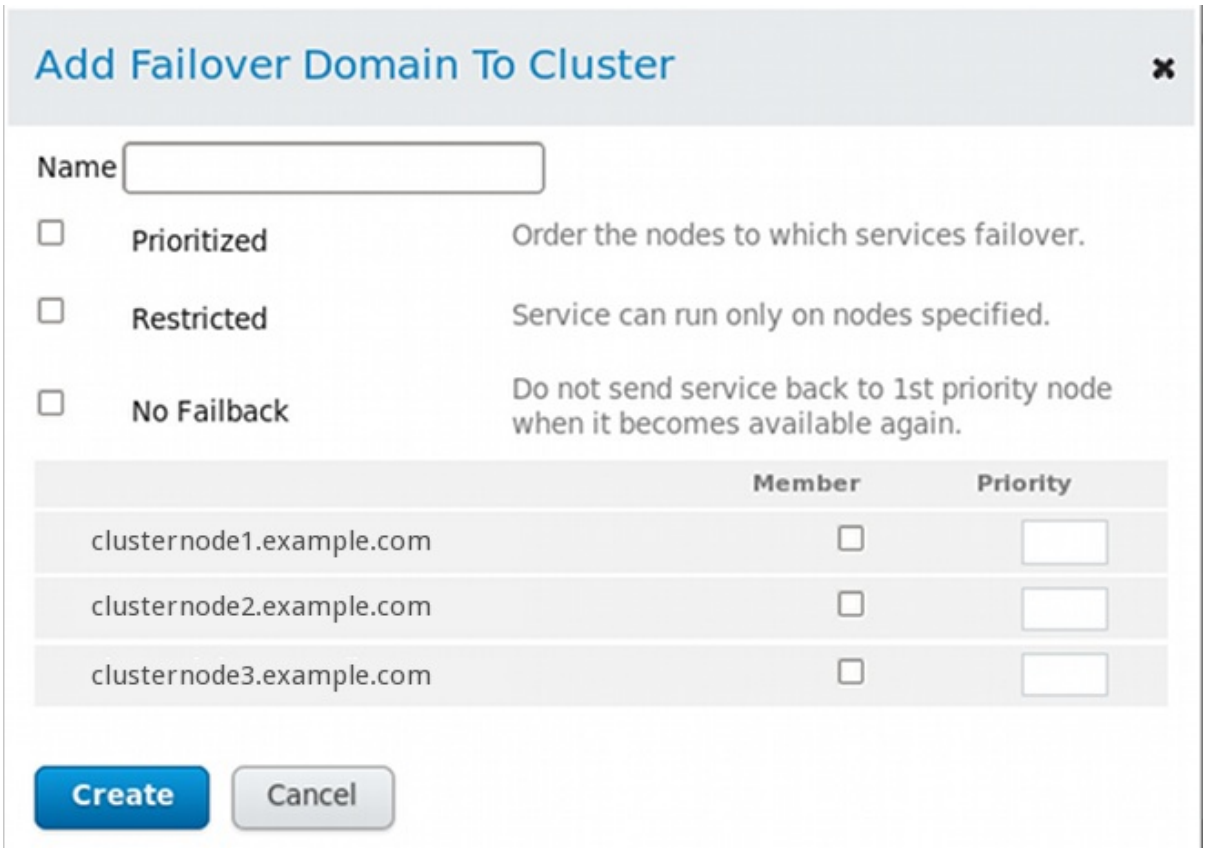


图 3.7. luci 故障切换域配置对话框

3. 在**「在集群中添加故障切换域」**对话框的**「名称」**文本框中指定故障切换域名称。

注意

该名称应该可以与集群中其它名称所显示的目的区别。

4. 要启用在故障切换域成员间设置故障切换优先权，请点击**「优先的」**复选框。选择**「优先的」**复选框后，您可以为选择作为故障切换域成员的每个节点设置优先值，**「优先权」**。

5. 要限制这个故障切换域成员的故障切换，请点击**「有限」**复选框。选择**「有限」**复选框后，分配给这个故障切换域的服务只能切换到这个故障切换域中的节点。

6. 要将那个节点指定为不在这个故障切换域中恢复，请点击**「无故障恢复」**复选框。选择**「无故障恢复」**后，如果从首选节点中恢复某个服务，则该服务不会切换到恢复它的节点中。

7. 配置这个故障切换域的成员。为每个要成为故障切换域成员的节点点击**「成员」**复选框。如果选择**「优先的」**复选框，则请为故障切换域每个成员在**「优先权」**文本框中设置优先权。

8. 点击 **创建** 按钮。此时会显示新创建故障切换域的**「故障切换域」**页面。出现一条信息显示创建了新的域。刷新该页面查看更新的状态。

3.8.2. 修改故障切换域

要修改故障切换域请按照本小节中的步骤执行。

1. 在具体集群页面中，您可以点击集群显示顶部的**「故障切换域」**为那个集群配置故障切换域。此时会显示为这个集群配置的故障切换域。

2. 点击故障切换域名称，此时会显示那个故障切换域的配置页面。

3. 要修改该故障切换域的**「优先」**、**「有限」**或者**「无故障切换恢复」**属性，请选择或者取消该属性旁的复选框，并点击 **更新属性** 按钮。

4. 要修改故障切换域成员，请选择或者取消集群成员旁的复选框。如果该故障切换域是有先的，您还可以为集群成员修改优先权设置。点击 **更新设置**。

3.8.3. 删除故障切换域

要删除故障切换域，请按照本小节中的步骤操作。

1. 在具体集群页面中，您可以点击集群显示顶部的**「故障切换域」**为那个集群配置故障切换域。此时会显示为这个集群配置的故障切换域。

2. 选择要删除的故障切换域前的复选框。

3. 点击**「删除」**。

3.9. 配置全局集群资源

您可以配置在集群中运行的任意服务所使用的全局资源，还可以配置只可用于具体服务的资源。

要添加全局集群资源，请按照本小节中的步骤操作。您可在配置该服务时，添加属于具体服务的本地资源，如 第 3.10 节 “在集群中添加集群服务” 所述。

1. 在具体集群页面中，您可点击集群显示顶部的**「资源」**菜单在那个集群中添加资源。此时会显示已经为那个集群添加的资源。

2. 点击**「添加」**。此时会显示**「在集群中添加资源」**下拉菜单。

3. 点击**「在集群中添加资源」**中的下拉框并选择要配置的资源类型。

4. 输入您要添加资源的资源参数。资源参数请参考 附录 B, *HA 资源参数*。

5. 点击 **提交**。点击 **提交** 按钮会返回显示**「资源」**信息的资源页面，此时该页面会显示添加的资源（和其它资源）。

要修改现有资源，请执行以下步骤。

1. 在 **luci** **「资源」**页面中点击要修改的资源名称。此时会显示那个资源的参数。

2. 编辑该资源的参数。

3. 点击 **应用**。

要删除现有资源，请执行以下步骤。

1. 在 **luci** **「资源」**页面中选择所有要删除资源。

2. 点击**「删除」**。

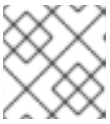
3.10. 在集群中添加集群服务

要在集群中添加集群服务，请按照本小节中的步骤执行。

1. 在具体集群页面中您可以点击集群显示顶部的**「服务组」**菜单在那个集群中添加服务。此时会显示已经为那个集群配置的服务。（在**「服务」**页面中，您还可以启动、重启以及禁用服务，如第 4.5 节 “管理高可用性服务” 所述。）

2. 点击**「添加」**。此时会显示**「在集群中添加服务组」**对话框。

3. 在**「在集群中添加服务组」**对话框的**「服务名称」**文本框中输入该服务名称。

   注意

   请使用可明确与集群中的其它服务区别开来的描述性名称。

4. 如果您想在启动并运行集群时自动启动该服务，请选择**「自动启动这个服务」**复选框。如果*没有*选择这个复选框，则您必须在集群不处于停止状态时手动启动该服务。

5. 选择**「独家运行」**复选框设置策略，即该服务只在没有其它服务运行的节点中运行。

6. 如果您已经为该集群配置了故障切换域，您可以使用**「故障切换域」**参数的下拉菜单为该服务选择故障切换域。有关故障切换域的详情请参考 第 3.8 节 “配置故障切换域”。

7. 使用**「恢复策略」**下拉框为该服务选择恢复策略。选项包括 **「重新定位」**、**「重启」**、**「重启-禁用」** 或者 **「禁用」**该服务。

   选择**「重启」**选项表示在重新定位该服务前系统应尝试重启失败的服务。选择**「重新定位」**选项表示系统应在不同节点中重启该服务。选择**「禁用」**选项表示如果任意组件失败，系统就应禁用该资源组。选择**「重启-禁用」**选项表示该服务失败的位置尝试重启该服务，但如果重启失败，则将禁用服务而不是移动到集群的另一台主机中。

   如果您选择**「重启」**或者**「重启-禁用」**作为该服务的恢复策略，您可以指定重新定位或者禁用该服务前最多重启失败的次数，您还可以在多少秒后不再重启。

8. 要在服务中添加资源，请点击 **添加资源**。点击 **添加资源** 按钮会显示一个 **在服务中添加资源** 下拉菜单，您可从中选择要添加的现有全局资源，或者添加一个*只可*用于这个服务的新资源。

   - 要添加现有全局资源，请在 **在服务中添加资源** 下拉框中点击现有资源名称。此时会显示在您所配置服务的**「服务组」**页面中的资源及其参数。有关添加或者修改全局资源的详情请参考 第 3.9 节 “配置全局集群资源”。

   - 要添加只可用于这个服务的新资源，请在 **在服务中添加资源** 下拉框中选择要配置的资源类型并为您要添加的资源输入资源参数。有关资源参数请参考 附录 B, *HA 资源参数*。

   - 当在服务中添加资源时，无论它是现有全局资源，还是只可用于这个服务的资源，您可将该资源指定为**「独立子树」**或者**「非关键资源」**。

     如果您将资源指定为独立子树，那么如果该资源失败，则在系统尝试常规恢复前只会重启那个资源（而不是整个服务）。您可以指定在该节点中为该服务使用恢复策略前最多尝试重启该资源的次数。您还可以指定在多少秒后系统将为该服务使用恢复策略。

     如果您将该资源指定为非关键资源，那么如果那个资源失败，则只需要重启该资源。同时如果该资源仍失败，那么只会禁用那个资源而不是整个服务。您可以指定在该节点中禁用该资源前最多重启该资源的次数。您还可以指定在多少秒后系统将禁用该资源。

9. 如果您要在您定义的资源中添加子资源，请点击 **添加子资源**。点击 **添加子资源** 后会显示**「在服务中添加资源」**下拉框，您可从中添加现有全局资源或者添加只可用于这个服务的新资源。您可以继续为这个资源添加子资源以适应您的要求。

注意

如果您要添加 Samba 服务资源，请将 Samba 服务资源直接连接到该服务，而*不是*服务中的资源。

10. 当您完成为该服务添加资源，并完成为资源添加子资源时，点击 **提交**。点击 **提交** 按钮后会返回显示添加的服务（以及其它服务）的**「服务组」**页面。

注意

要确认在集群服务中使用的 IP 服务资源，您可以在集群节点中使用 **/sbin/ip addr show** 命令而不是弃用的 **ifconfig** 命令。以下显示了在运行集群服务的节点中运行 **/sbin/ip addr show** 的输出结果：

```
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1356 qdisc pfifo_fast
qlen 1000
    link/ether 00:05:5d:9a:d8:91 brd ff:ff:ff:ff:ff:ff
    inet 10.11.4.31/22 brd 10.11.7.255 scope global eth0
    inet6 fe80::205:5dff:fe9a:d891/64 scope link
    inet 10.11.4.240/22 scope global secondary eth0
       valid_lft forever preferred_lft forever
```

要修改现有服务，请执行以下步骤。

1. 在**「服务组」**对话框中点击要修改的服务名称。此时会显示您已经为那个服务配置的参数和资源。
2. 编辑该服务的参数。
3. 点击 **提交**。

要删除一个或多个现有资源，请执行以下步骤。

1. 在 **luci「服务组」**页面中选择要删除的所有服务。
2. 点击**「删除」**。
3. 从红帽企业版 Linux 6.3 开始，在从 **luci** 中删除任意服务前会出现一条信息，询问您是否确定要删除该服务组或多个服务组，这样会停止组成该组的服务。点击**「取消」**关闭该对话框而不会删除任何服务，或者点击**「处理」**删除所选服务或多个服务。

第 4 章 使用 CONGA 管理 RED HAT 高可用性附加组件

本章论述了管理 Red Hat 高可用性附加组件的各种管理任务，它由以下小节组成：

4.1. 在 LUCI 界面中添加现有集群

如果您之前已创建了高可用性附加组件集群，则可以在 **luci** 界面中轻松添加该集群，以便使用 **Conga** 管理该集群。

要在 **luci** 界面中添加现有集群，请执行以下步骤：

1. 在 **luci**「**Homebase**」页面左侧的菜单中点击「**管理**」。此时会出现「**集群**」页面。
2. 点击「**添加**」。此时会出现「**添加现有集群**」页面。
3. 在现有集群的所有节点中输入节点主机名和 **ricci** 密码。因为该集群的每个节点中都有关于该集群的所有配置信息，这样就应该提供了在 **luci** 界面中添加集群的足够信息。
4. 点击 **连接**。此时「**添加现有集群**」页面会显示集群名称以及该集群中的其他节点。
5. 为该集群的每个节点输入独立的 **ricci** 密码，或者输入一个密码并选择「**在所有节点中使用同一密码**」。
6. 点击 **添加集群**。此时会在「**管理集群**」页面中出现之前配置的集群。

4.2. 从 LUCI 界面中删除一个集群

您可以从 **luci** 管理 GUI 中删除集群而不影响集群服务或集群成员。如果要删除一个集群，还可随后添加回来，或者您可以将其添加到另一个 **luci** 事务中，如 第 4.1 节 “在 luci 界面中添加现有集群” 所示。

要从 **luci** 管理 GUI界面中删除集群而不影响集群服务或集群成员，请按以下步骤操作：

1. 在 **luci**「**Homebase**」页面左侧的菜单中点击「**管理**」。此时会出现「**集群**」页面。
2. 选择要删除的集群或多个集群。
3. 点击「**删除**」。

有关删除整个集群，停止所有集群服务，在节点中删除该集群配置信息的详情请参考 第 4.4 节 “启动、停止、刷新和删除集群”。

4.3. 管理集群节点

本小节论述了如何使用 **Conga** 的 **luci** 服务器组件执行以下节点管理功能：

- 第 4.3.1 节 “重启集群节点”
- 第 4.3.2 节 “使节点离开或者加入集群”
- 第 4.3.3 节 “在运行的集群中添加成员”
- 第 4.3.4 节 “删除集群中的成员”

4.3.1. 重启集群节点

要在集群中重启节点，请执行以下步骤：

1. 在具体集群页面中点击集群显示顶端的**「节点」**，此时会显示组成该集群的节点。这也是您点击 **luci**「**Homebase**」页面左侧菜单中集群名称下的**「管理集群」**时出现的默认页面。
2. 点击要重启的节点旁的复选框选择该节点。
3. 选择该页面顶端菜单中的**「重启」**功能。此时会重启所选节点，并在该页面顶部出现一条信息表示正在重启该节点。
4. 刷新该页面查看该节点更新的状态。

您还可以在点击**「重启」**前选择所有要重启的节点，这样可以一次重启多个节点。

4.3.2. 使节点离开或者加入集群

您可以使用 **Conga** 的 **luci** 服务器组件，通过停止该节点中的所有集群服务，让该节点离开集群。您还可以使用 **Conga** 的 **luci** 服务器组件让已经离开的节点重新加入该集群。

让集群离开节点并不会从该节点中删除集群配置信息，且该节点仍会出现在该集群节点显示中，只是状态为**不是集群成**员。有关从集群配置中完全删除节点的信息请参考 第 4.3.4 节 “删除集群中的成员”。

要让节点离开集群，请执行以下步骤，这样可关闭该节点中的集群软件。让节点离开集群可防止在重启时该节点自动加入集群。

1. 在具体集群页面中点击集群显示顶端的**「节点」**，此时会显示组成该集群的节点。这也是您点击 **luci**「**Homebase**」页面左侧菜单中集群名称下的**「管理集群」**时出现的默认页面。
2. 点击节点旁的复选框选择您想要使其离开集群的节点。
3. 在该页面顶部菜单中选择**「离开集群」**功能，此时会在页面顶部出现一条信息表明已经停止该节点。
4. 刷新该页面查看该节点更新的状态。

您还可以在点击**「离开节点」**前选择所有您想要使其离开集群的节点，这样可一次让多个节点离开。

要让节点重新加入集群，请那些节点旁的复选框选择您想要重新加入到集群的节点并选择**「加入集群」**。这样可让选择的节点加入集群，并使其在重启时加入集群。

4.3.3. 在运行的集群中添加成员

要在运行的集群中添加成员，请按照本小节中的步骤执行。

1. 在具体集群页面中点击集群显示顶部的**「节点」**，此时会显示组成该集群的节点。这也是您点击 **luci「Homebase」**页面左侧菜单中集群名称下的**「管理集群」**菜单时默认出现的页面。

2. 点击**「添加」**。点击**「添加」**会显示**「在集群中添加节点」**对话框。

3. 在**「节点主机名」**文本框中输入节点的名称，在**「密码」**文本框中输入 **ricci** 密码。如果您要在 **ricci** 代理中使用不同于默认 11111 的端口，您可以将该参数改为您正在使用的端口。

4. 如果需要集群的存储，请选择**「启动共享存储支持」**复选框，下载支持集群存储的软件包，并启用集群的 LVM。您应该只在能够访问弹性存储附加组件或者可扩展文件系统附加组件时选择这个选项。

5. 如果您要添加多个节点，请点击 **添加另一个节点**，并为每个附加节点 输入节点名称和密码。

6. 点击 **添加节点**。点击 **添加节点** 会导致以下动作：

    1. 如果您选择**「下载软件包」**，则会在节点中下载集群软件包。

    2. 在节点中安装集群软件（或者确认安装了正确的软件包）。

    3. 更新集群配置，并在集群的每个节点中使用更新的集群配置 — 包括添加的节点。

    4. 添加的节点加入集群。

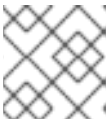
    「节点」页面出现一条信息表示正在该集群中添加节点。刷新该页面更新状态。

7. 当添加节点进程完成后，点击新添加的节点名称为这个节点配置 fencing，如 第 3.6 节 “配置 Fence 设备” 所述。

4.3.4. 删除集群中的成员

要从目前处于操作状态的现有集群中删除成员，请按照本小节中的步骤执行。请注意：必须在删除节点前停止它们，除非您要同时删除该集群中的所有节点。

1. 在具体集群页面中点击集群显示顶部的**「节点」**，此时会显示组成该集群的节点。这也是您点击 **luci「Homebase」**页面左侧菜单中集群名称下的**「管理集群」**菜单时默认出现的页面。



注意

要在删除节点时让所有在该节点中运行的服务执行故障切换，请跳过下一步。

2. 禁用或者重新定位所要删除节点中运行的所有服务。有关禁用和重新定位服务的详情请参考 第 4.5 节 “管理高可用性服务”。

3. 选择要删除的节点。

4. 点击**「删除」**。**「节点」**页面显示正在删除该节点。刷新该页面查看当前状态。

重要

从集群中删除集群节点是破坏性操作，不能撤销。

4.4. 启动、停止、刷新和删除集群

您可以通过在集群的每个节点中执行以下动作启动、停止、重启某个集群。在具体集群页面中点击集群显示顶部的**「节点」**，此时会显示组成该集群的节点。

如果要将集群服务移动到另一个集群成员中，则在集群节点或整个集群中执行启动和重启操作时会造成短暂的集群服务中断，因为它是在要停止或重启的节点中运行。

要停止集群，请执行以下步骤。这样会关闭节点中的集群软件，但不会从节点中删除集群配置信息，且该节点仍会出现在该集群节点显示中，只是状态为**不是集群成员**。

1. 点击每个节点旁的复选框选择集群中的所有节点。
2. 在该页面顶部的菜单中选择**「离开集群」**，此时会在页面顶部出现一条信息表示正在停止每个节点。
3. 刷新该页面查看节点更新的状态。

要启动集群，请执行以下步骤：

1. 点击每个节点旁的复选框选择集群中的所有节点。
2. 在该页面顶部的菜单中选择**「加入集群」**功能。
3. 刷新该页面查看节点更新的状态。

要重启运行中的集群，首先请停止集群中的所有节点，然后启动集群中的所有节点，如上所述。

要删除整个集群，请按照以下步骤执行。这导致所有集群服务停止，并从节点中删除该集群配置信息，同时在集群显示中删除它们。如果您之后尝试使用已删除的节点添加现有集群，**luci** 将显示该节点已不是任何集群的成员。

重要

删除集群是一个破坏性操作，且无法撤销。要在删除集群后进行恢复，您需要从头开始重新创建并重新定义该集群。

1. 点击每个节点旁的复选框选择集群中的所有节点。
2. 在该页面顶部的菜单中选择**「删除」**功能。

如果您要从 **luci** 界面中删除某个集群而不停止任何集群服务或者更改集群成员属性，您可以使用**「管理集群」**页面中的**「删除」**选项，如 第 4.2 节 “从 luci 界面中删除一个集群” 所示。

4.5. 管理高可用性服务

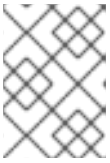
除在 第 3.10 节 “在集群中添加集群服务” 中所述的添加和修改服务外，您还可以使用 **Conga** 的 **luci** 服务器组件为高可用性服务执行以下管理功能：

- 启动服务
- 重启服务
- 禁用服务
- 删除服务
- 重新定位服务

在具体集群页面中，您可以点击集群显示顶部的**「服务组」**为集群管理服务。此时会显示为该集群配置的服务。

- **「启动服务」** — 要启动任何当前没有运行的服务，请点击该服务旁的复选框选择您要启动的所有服务，并点击**「启动」**。
- **「重启服务」** — 要重启任何当前运行的服务，请点击该服务旁的复选框选择您要启动的所有服务，并点击**「重启」**。
- **「禁用服务」** — 要禁用任何当前运行的服务，请点击该服务旁的复选框选择您要启动的所有服务并，点击**「禁用」**。
- **「删除服务」** — 要删除任何当前运行的服务，请点击该服务旁的复选框选择您要启动的所有服务并点击**「删除」**。
- **「重新定位服务」** — 要重新定位运行的服务，请在服务显示中点击该服务的名称。此时会显示该服务配置页面，并显示该服务目前在哪个节点中运行。

  在**「在节点中启动.....」**下拉框中选择您想要将服务重新定位的节点，并点击**「启动」**图标。此时会在页面顶部显示一条信息说明正在重启该服务。您可以刷新该页面查看新显示，在该显示中说明该服务正在您选择的节点中运行。



注意

如果您所选运行的服务是一个 **vm** 服务，下拉框中将会显示 **migrate** 选项而不是 **relocate** 选项。

注意

您还可以点击**「服务」**页面中的服务名称启动、重启、禁用或者删除独立服务。此时会显示服务配置页面。在服务配置页面右上角有一些图标：**「启动」**、**「重启」**、**「禁用」**和**「删除」**。

4.6. 备份和恢复 LUCI 配置

从红帽企业版 Linux 6.2 开始，您可以使用以下步骤备份 **luci** 数据库，即保存在 **/var/lib/luci/data/luci.db** 文件中。这不是给集群自身的配置，自身配置保存在 **cluster.conf** 文件中。相反，它包含用户和集群以及 **luci** 维护的相关属性列表。默认情况下，备份生成的步骤将会写入同一目录的 **luci.db** 文件中。

1. 执行 **service luci stop**。
2. 执行 **service luci backup-db**。

   您可以选择是否指定一个文件名作为 **backup-db** 命令的参数，该命令可将 **luci** 数据库写入那个文件。例如：要将 **luci** 数据库写入文件 **/root/luci.db.backup**，您可以执行命令 **service luci backup-db /root/luci.db.backup**。注：但如果将备份文件写入 **/var/lib/luci/data/** 以外的位置（您使用 **service luci backup-db** 指定的备份文件名）将不会在 **list-backups** 命令的输出结果中显示。
3. 执行 **service luci start**。

使用以下步骤恢复 **luci** 数据库。

1. 执行 **service luci stop**。

2. 执行 **service luci list-backups**，并注释要恢复的文件名。

3. 执行 **service luci restore-db /var/lib/luci/data/*lucibackupfile***，其中 *lucibackupfile* 是要恢复的备份文件。

   例如：以下命令恢复保存在备份文件 **luci-backup20110923062526.db** 中的 **luci** 配置信息：

   ```
   service luci restore-db /var/lib/luci/data/luci-
   backup20110923062526.db
   ```

4. 执行 **service luci start**。

如果您需要恢复 **luci** 数据库，但在您因完全重新安装生成备份的机器中已丢失 **host.pem** 文件，例如：您需要将集群重新手动添加回 **luci** 方可重新认证集群节点。

请使用以下步骤在生成备份之外的机器中恢复 **luci** 数据库。注：除恢复数据库本身外，您还需要复制 SSL 证书文件，以保证在 **ricci** 节点中认证 **luci**。在这个示例中是在 **luci1** 机器中生成备份，在 **luci2** 机器中恢复备份。

1. 执行以下一组命令在 **luci1** 中生成 **luci** 备份，并将 SSL 证书和 **luci** 备份复制到 **luci2** 中。

   ```
   [root@luci1 ~]# service luci stop
   [root@luci1 ~]# service luci backup-db
   [root@luci1 ~]# service luci list-backups
   /var/lib/luci/data/luci-backup20120504134051.db
   [root@luci1 ~]# scp /var/lib/luci/certs/host.pem
   /var/lib/luci/data/luci-backup20120504134051.db root@luci2:
   ```

2. 在 **luci2** 机器中，保证已安装 **luci**，且没有运行。如果还没有安装，则请安装该软件包。

3. 执行以下一组命令保证认证到位，并在 **luci2** 中使用 **luci1** 恢复 **luci** 数据库。

   ```
   [root@luci2 ~]# cp host.pem /var/lib/luci/certs/
   [root@luci2 ~]# chown luci: /var/lib/luci/certs/host.pem
   [root@luci2 ~]# /etc/init.d/luci restore-db ~/luci-
   backup20120504134051.db
   [root@luci2 ~]# shred -u ~/host.pem ~/luci-backup20120504134051.db
   [root@luci2 ~]# service luci start
   ```

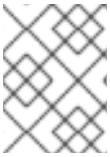
第 5 章 使用 CCS 命令配置红帽高可用性附加组件

从红帽企业版 Linux 6.1 开始，红帽高可用性附加组件支持 **ccs** 集群配置命令。**ccs** 命令可让管理员创建、修改和查看 **cluster.conf** 集群配置文件。您可以使用 **ccs** 命令在本地文件系统，或者远程节点中配置集群配置文件。管理员还可以使用 **ccs** 命令在配置的集群的一个或者全部节点中启动或者停止集群服务。

本章论述了如何使用 **ccs** 命令配置红帽高可用性附加组件集群配置文件。有关使用 **ccs** 命令管理运行的集群的详情请参考 第 6 章 *使用 **ccs** 管理 Red Hat 高可用性附加组件*。

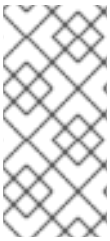
本章由以下小节组成：

- 第 5.1 节 “操作概述”
- 第 5.2 节 “配置任务”
- 第 5.3 节 “启动 **ricci**”
- 第 5.4 节 “创建集群”
- 第 5.5 节 “配置 Fence 设备”
- 第 5.7 节 “为集群成员配置 Fencing”
- 第 5.8 节 “配置故障切换域”
- 第 5.9 节 “配置全局集群资源”
- 第 5.10 节 “在集群中添加集群服务”
- 第 5.13 节 “配置仲裁磁盘”
- 第 5.14 节 “其他集群配置”
- 第 5.14 节 “其他集群配置”
- 第 5.15 节 “在集群节点中推广配置文件”



注意

请确定您部署的高可用性附加组件符合您的要求，且可被支持。请在部署前咨询授权红帽代表确认您的配置。另外还要预留一定的时间进行配置的失败模型测试。



注意

本章通常使用 **cluster.conf** 元素和属性作为参考。有关 **cluster.conf** 元素和属性的完整列表，请参考 **/usr/share/cluster/cluster.rng** 中的集群方案以及 **/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html** 中有注释的方案（例如：**/usr/share/doc/cman-3.0.12/cluster_conf.html**）。

5.1. 操作概述

本小节论述了以下使用 **ccs** 命令配置集群的常规操作：

- 第 5.1.1 节 “在本地系统中创建集群配置文件”

- 第 5.1.2 节 “查看当前集群配置”
- 第 5.1.3 节 “使用 ccs 命令指定 ricci 密码”
- 第 5.1.4 节 “修改集群配置组件”

5.1.1. 在本地系统中创建集群配置文件

使用 **ccs** 命令，您可以在集群节点中创建集群配置文件，或者在本地文件系统中创建集群配置文件，然后将其发送到集群的主机中。这可让您访问本地机器中的文件，使用版本控制维护该文件，或者根据您的需要标记该文件。使用 **ccs** 命令不需要 root 特权。

当您在集群节点中使用 **ccs** 命令创建并编辑集群配置文件时，请使用 **-h** 选项指定主机名。这样就可在该主机中创建并编辑 **cluster.conf** 文件：

```
ccs -h host [options]
```

要在本地系统中创建并编辑集群配置文件，请使用 **ccs** 命令的 **-f** 选项指定在执行集群操作时的配置文件名称。您可以使用任意名称命名该文件。

```
ccs -f file [options]
```

您在本地创建该文件后，可使用 **ccs** 命令的 **--setconf** 选项将其发送到集群节点中。在集群主机中会将您发送的文件命名为 **cluster.conf**，并将其保存在 **/etc/cluster** 目录中。

```
ccs -h host -f file --setconf
```

有关使用 **ccs** 命令的 **--setconf** 选项的详情请参考 第 5.15 节 “在集群节点中推广配置文件”。

5.1.2. 查看当前集群配置

如果在创建集群配置文件的过程中要打印当前文件，可使用以下命令，指定该集群中的某个节点作为主机：

```
ccs -h host --getconf
```

如果您要在本地系统中创建了您的配置文件，可以使用 **-f** 选项而不是 **-h** 选项，如 第 5.1.1 节 “在本地系统中创建集群配置文件” 所述。

5.1.3. 使用 ccs 命令指定 ricci 密码

执行 **ccs** 命令在集群的节点中发布 **cluster.conf** 文件的副本需要在该集群节点中安装并运行 **ricci**，如 第 2.13 节 “**ricci** 注意事项” 所述。使用 **ricci** 时，第一次在任意具体机器中与 **ricci** 互动时要求输入密码。

如果您还没有在您正使用的机器中为具体机器中的 **ricci** 事务输入密码，在会在 是要 **ccs** 命令需要密码时提示您输入。您也可以使用 **-p** 选项在命令行中指定 **ricci** 密码。

```
ccs -h host -p password --sync --activate
```

当您在集群的所有节点中使用 **ccs** 命令的 **--sync** 选项推广 **cluster.conf** 文件， 并为该命令指定 **ricci** 密码时，**ccs** 命令将在该集群的每个节点中使用那个密码。如果您需要为 **ricci** 在独立节点中设定不同密码，您可以同时使用 **--setconf** 选项和 **-p** 选项，每次在一个节点中发布配置文件。

5.1.4. 修改集群配置组件

您使用 **ccs** 命令配置集群配置文件中的集群组件及其属性。您在该文件中添加集群组件后，要修改那个组件的属性，就必须删除已定义的组件，并使用修改的属性再次添加该组件。本章的各个小节提供了如何对每个组件进行此操作的信息。

cman 集群组件属性为修改集群组件的步骤提供一个例外。要修改这些属性，请执行附带 **--setcman** 选项的 **ccs** 命令，指定新的属性。注：指定这个选项会重置所有您没有为其默认值特别指定的所有值，如第 5.1.5 节 “覆盖之前设置的命令” 所述。

5.1.5. 覆盖之前设置的命令

设定属性时，**ccs** 命令使用的一些选项会覆盖警告。这意味着您可以在 **ccs** 命令中使用这些选项之一而无需指定任何设置，同时它会重新将所有设置恢复到其默认值。这些选项如下：

- **--settotem**
- **--setdlm**
- **--setrm**
- **--setcman**
- **--setmulticast**
- **--setaltmulticast**
- **--setfencedaemon**
- **--setlogging**
- **--setquorumd**

例如：要重新设置该 fence 守护进程的所有属性，您可以运行以下命令：

```
# ccs -h hostname --setfencedaemon
```

注：如果您使用这些命令之一重置属性，那么该命令的其他属性会恢复到其默认值。例如：如果使用以下命令将 **post_fail_delay** 属性设定为 5：

```
# ccs -h hostname --setfencedaemon post_fail_delay=5
```

如果运行那个命令后，您运行以下命令将 **post_join_delay** 属性重新设定为 10，**post_fail_delay** 属性将恢复到其默认值：

```
# ccs -h hostname --setfencedaemon post_join_delay=10
```

要重置 **post_fail_delay** 和 **post_join_delay** 属性，您可以在同一命令中同时指定他们，如下示例所示：

```
# ccs -h hostname --setfencedaemon post_fail_delay=5 post_join_delay=10
```

有关配置 fence 设备的详情请参考 第 5.5 节 “配置 Fence 设备”。

5.1.6. 配置验证

当您使用 **ccs** 命令生成并编辑集群配置文件时，会根据集群方案自动验证该配置。从红帽企业版 Linux 6.3 开始，**ccs** 命令根据您使用 **-h** 选项所指定节点中位于 **/usr/share/cluster/cluster.rng** 文件的集群方案验证配置。之前 **ccs** 命令总是使用打包在 **ccs** 命令中的集群方案，即本地系统中的 **/usr/share/ccs/cluster.rng**。当您使用 **-f** 选项指定本地系统时，**ccs** 命令仍使用那个系统中 **ccs** 自身拥有的集群方案 **/usr/share/ccs/cluster.rng**。

5.2. 配置任务

使用 **ccs** 配置红帽高可用性附加组件软件包括以下步骤：

1. 确定在该集群的所有节点中运行 **ricci**，请参考 第 5.3 节 “启动 **ricci**”。
2. 创建集群。请参考 第 5.4 节 “创建集群”。
3. 配置 fence 设备。请参考 第 5.5 节 “配置 Fence 设备”。
4. 为集群成员配置 fencing。请参考 第 5.7 节 “为集群成员配置 Fencing”。
5. 创建故障切换域。请参考 第 5.8 节 “配置故障切换域”。
6. 创建资源。请参考 第 5.9 节 “配置全局集群资源”。
7. 创建集群服务。请参考 第 5.10 节 “在集群中添加集群服务”。
8. 需要时配置仲裁磁盘。请参考 第 5.13 节 “配置仲裁磁盘”。
9. 配置全局集群属性。请参考 第 5.14 节 “其他集群配置”。
10. 将该集群配置文件传播到所有集群节点中。请参考 第 5.15 节 “在集群节点中推广配置文件”。

5.3. 启动 RICCI

要在该集群的节点中创建并部署集群配置文件，必须在每个节点中运行 **ricci** 服务。启动 **ricci** 前，您应该确定您将系统配置为满足如下要求：

1. 应在您的集群节点中为 **ricci** 启用 IP 端口。有关在集群节点中启用 IP 端口的详情请参考 第 2.3.1 节 “在集群节点中启用 IP 端口”。
2. 在该集群的所有节点中安装 **ricci** 服务，并分配 **ricci** 密码，如 第 2.13 节 “**ricci** 注意事项” 所述。

在每个节点中安装并配置 **ricci** 后，在每个节点中启动 **ricci** 服务：

```
# service ricci start
Starting ricci:                                            [  OK  ]
```

5.4. 创建集群

本小节论述了如何使用 **ccs** 命令，在没有 fencing、故障切换域和 HA 服务的情况下创建、修改并删除集群配置框架。随后的小节论述了如何设置配置文件的那些部分。

要创建集群配置文件框架，请首先创建并命名该集群，然后在该集群中添加节点，如下所示：

1. 在该集群中的节点中执行 **ccs** 命令创建集群配置文件，使用 **-h** 参数指定创建该文件的节点，并使用 **createcluster** 选项指定该集群名称：

   ```
   ccs -h host --createcluster clustername
   ```

   例如：下面的命令在 **node-01.example.com** 中创建了名为 **mycluster** 的配置文件：

   ```
   ccs -h node-01.example.com --createcluster mycluster
   ```

   集群名称不能超过 15 个字符。

   如果您指定的主机中已经存在 **cluster.conf** 文件，执行这个命令将替换现有文件。

   如果您要在本地系统中创建集群配置文件，可以使用 **-f** 选项而不是 **-h** 选项。有关在本地创建该文件的详情请参考 第 5.1.1 节 “在本地系统中创建集群配置文件” 所述。

2. 要配置该集群包含的节点，请在该集群的每个节点中执行以下命令：

   ```
   ccs -h host --addnode node
   ```

   例如：以下三个命令可在 **node-01.example.com** 的配置文件中添加 **node-01.example.com**、**node-02.example.com** 和 **node-03.example.com**。

   ```
   ccs -h node-01.example.com --addnode node-01.example.com
   ccs -h node-01.example.com --addnode node-02.example.com
   ccs -h node-01.example.com --addnode node-03.example.com
   ```

   要查看集群中已经配置的节点列表，请执行以下命令：

   ```
   ccs -h host --lsnodes
   ```

   例 5.1 “**cluster.conf** File After Adding Three Nodes” 演示了您创建了包括节点 **node-01.example.com**、**node-02.example.com** 和 **node-03.example.com** 的集群 **mycluster** 后的 **cluster.conf** 配置文件。

   例 5.1. cluster.conf File After Adding Three Nodes

   ```
   <cluster name="mycluster" config_version="2">
      <clusternodes>
        <clusternode name="node-01.example.com" nodeid="1">
            <fence>
            </fence>
        </clusternode>
        <clusternode name="node-02.example.com" nodeid="2">
            <fence>
            </fence>
        </clusternode>
   ```

```
      <clusternode name="node-03.example.com" nodeid="3">
          <fence>
          </fence>
      </clusternode>
    </clusternodes>
    <fencedevices>
    </fencedevices>
    <rm>
    </rm>
</cluster>
```

当您在该集群中添加节点时，您可以指定该节点所供献的用来确定是否合法的票数。请使用以下命令为集群节点设定票数：

```
ccs -h host --addnode host --votes votes
```

添加节点时，**ccs** 会为该节点分配一个唯一证书作为该节点的识别符。如果您要在创建节点时手动指定节点识别符，请使用以下命令：

```
ccs -h host --addnode host --nodeid nodeid
```

要从集群中删除节点，请执行以下命令：

```
ccs -h host --rmnode node
```

您完成配置集群的所有组件后，需要在所有节点中同步该集群配置文件，如 第 5.15 节 “在集群节点中推广配置文件” 所述。

5.5. 配置 FENCE 设备

配置 fence 设备包括为集群创建、更新和删除 fence 设备。在您可以在集群中为节点配置 fencing 前必须在集群中创建并命名 fence 设备。有关为该集群中的独立节点配置 fencing 的详情请参考 第 5.7 节 “为集群成员配置 Fencing”。

在配置 fence 设备前，您可能想修改系统 fence 守护进程属性的默认值。为 fence 守护进程配置的值应该是集群的常规值。您可能想要为集群修改的常规 fencing 属性如下：

- **post_fail_delay** 属性是在节点失败后，fencing 节点前 fence 守护进程（**fenced**）要等待的秒数。**post_fail_delay** 的默认值为 **0**。可使用不同的数值以适应集群和网络性能。
- **post-join_delay** 属性是该节点加入 fence 守护进程（**fenced**）后，该守护进程 fence 该节点前要等待的秒数。**post-join_delay** 默认值为 **6**。**post-join_delay** 一般在 20-30 秒之间，可根据集群和网络性能而有所不同。

您可以使用 **ccs** 命令的 **--setfencedaemon** 选项重新设定 **post_fail_delay** 和 **post_join_delay** 的值。注：执行 **ccs --setfencedaemon** 命令将覆盖现有特别设定的 fence 守护进程属性，将其恢复到默认值。

例如：要配置 **post_fail_delay** 属性值，请执行以下命令。这个命令将覆盖您已经使用这个命令设定的其他所有现有 fence 守护进程属性，并将其恢复到默认值。

```
ccs -h host --setfencedaemon post_fail_delay=value
```

要配置 **post_join_delay** 属性值，请执行以下命令。这个命令将覆盖您已经使用这个命令设定的其他所有现有 fence 守护进程属性，并将其恢复到默认值。

```
ccs -h host --setfencedaemon post_join_delay=value
```

请执行以下命令同时为 **post_join_delay** 属性和 **post_fail_delay** 属性配置属性值：

```
ccs -h host --setfencedaemon post_fail_delay=value post_join_delay=value
```

注意

有关 **post_join_delay** 和 **post_fail_delay** 属性，以及您可以修改的附加 fence 守护进程的详情请参考 fenced(8) man page，**/usr/share/cluster/cluster.rng** 中的集群方案以及 **/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html** 中注释的方案。

请执行以下命令为集群配置 fence 设备：

```
ccs -h host --addfencedev devicename [fencedeviceoptions]
```

例如：执行以下命令在集群节点 **node1** 的配置文件中配置一个 APC fence 设备，名为 **myfence**，IP 地址为 **apc_ip_example**，登录为 **login_example**，密码为 **password_example**：

```
ccs -h node1 --addfencedev myfence agent=fence_apc ipaddr=apc_ip_example
login=login_example passwd=password_example
```

下面的示例演示了添加这个 APC fence 设备后 **cluster.conf** 配置文件的 **fencedevices** 部分：

```
<fencedevices>
      <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="myfence" passwd="password_example"/>
</fencedevices>
```

为集群配置 fence 设备时，您可能会发现查看您集群的可用设备列表以及每个设备的可用选项会有所帮助。您还可能会发现查看目前为您集群配置的 fence 设备列表也有帮助。有关使用 **ccs** 命令列出可用 fence 设备及选项，或者输出目前为您集群配置的 fence 设备列表的详情，请参考 第 5.6 节 “列出 Fence 设备和 Fence 设备选项”。

请执行以下命令从您的集群配置中删除 fence 设备：

```
ccs -h host --rmfencedev fence_device_name
```

例如：执行以下命令从集群节点 **node1** 的集群配置文件中删除名为 **myfence** 的 fence 设备：

```
ccs -h node1 --rmfencedev myfence
```

如果您要修改已经配置的 fence 设备的属性，必须首先删除那个 fence 设备，然后使用修改的属性再次添加该设备。

注：完成集群的所有组件配置后，需要在所有节点中同步该集群配置文件，如 第 5.15 节 “在集群节点中推广配置文件” 所述。

5.6. 列出 FENCE 设备和 FENCE 设备选项

您可以使用 **ccs** 命令输出可用 fence 设备列表，并列出每个可用 fence 类型的选项列表。您还可以使用 **ccs** 命令输出目前为您集群配置的 fence 设备列表。

请执行以下命令输出目前为您集群配置的 fence 设备列表：

```
ccs -h host --lsfenceopts
```

例如：以下命令列出集群节点 **node1** 中可用的 fence 设备，演示示例输出结果：

```
[root@ask-03 ~]# ccs -h node1 --lsfenceopts
fence_rps10 - RPS10 Serial Switch
fence_vixel - No description available
fence_egenera - No description available
fence_xcat - No description available
fence_na - Node Assassin
fence_apc - Fence agent for APC over telnet/ssh
fence_apc_snmp - Fence agent for APC over SNMP
fence_bladecenter - Fence agent for IBM BladeCenter
fence_bladecenter_snmp - Fence agent for IBM BladeCenter over SNMP
fence_cisco_mds - Fence agent for Cisco MDS
fence_cisco_ucs - Fence agent for Cisco UCS
fence_drac5 - Fence agent for Dell DRAC CMC/5
fence_eps - Fence agent for ePowerSwitch
fence_ibmblade - Fence agent for IBM BladeCenter over SNMP
fence_ifmib - Fence agent for IF MIB
fence_ilo - Fence agent for HP iLO
fence_ilo_mp - Fence agent for HP iLO MP
fence_intelmodular - Fence agent for Intel Modular
fence_ipmilan - Fence agent for IPMI over LAN
fence_kdump - Fence agent for use with kdump
fence_rhevm - Fence agent for RHEV-M REST API
fence_rsa - Fence agent for IBM RSA
fence_sanbox2 - Fence agent for QLogic SANBox2 FC switches
fence_scsi - fence agent for SCSI-3 persistent reservations
fence_virsh - Fence agent for virsh
fence_virt - Fence agent for virtual machines
fence_vmware - Fence agent for VMware
fence_vmware_soap - Fence agent for VMware over SOAP API
fence_wti - Fence agent for WTI
fence_xvm - Fence agent for virtual machines
```

请执行以下命令列出您可为具体 fence 类型指定的选项列表：

```
ccs -h host --lsfenceopts fence_type
```

例如：下面的命令列出 **fence_wti** fence 代理的 fence 选项列表。

```
[root@ask-03 ~]# ccs -h node1 --lsfenceopts fence_wti
fence_wti - Fence agent for WTI
  Required Options:
  Optional Options:
    option: No description available
    action: Fencing Action
    ipaddr: IP Address or Hostname
    login: Login Name
    passwd: Login password or passphrase
    passwd_script: Script to retrieve password
    cmd_prompt: Force command prompt
    secure: SSH connection
    identity_file: Identity file for ssh
    port: Physical plug number or name of virtual machine
    inet4_only: Forces agent to use IPv4 addresses only
    inet6_only: Forces agent to use IPv6 addresses only
    ipport: TCP port to use for connection with device
    verbose: Verbose mode
    debug: Write debug information to given file
    version: Display version information and exit
    help: Display help and exit
    separator: Separator for CSV created by operation list
    power_timeout: Test X seconds for status change after ON/OFF
    shell_timeout: Wait X seconds for cmd prompt after issuing command
    login_timeout: Wait X seconds for cmd prompt after login
    power_wait: Wait X seconds after issuing ON/OFF
    delay: Wait X seconds before fencing is started
    retry_on: Count of attempts to retry power on
```

请执行以下命令输出当前为您的集群配置的 fence 设备列表：

```
ccs -h host --lsfencedev
```

5.7. 为集群成员配置 FENCING

完成创建集群以及创建 fence 设备的初始步骤后，需要为集群节点配置 fencing。请按照本小节提供的步骤在创建新集群并为该集群配置 fencing 设备后，为节点配置 fencing。注：必须为该集群中的每个节点配置 fencing。

本小节记录了以下步骤：

- 第 5.7.1 节 “为节点配置使用单一电源的 Fence 设备”
- 第 5.7.2 节 “为节点配置单一存储 Fence 设备”
- 第 5.7.3 节 “配置备用 Fence 设备”
- 第 5.7.4 节 “配置使用冗余电源的节点”
- 第 5.7.5 节 “删除 Fence 方法和 Fence 事务”

5.7.1. 为节点配置使用单一电源的 Fence 设备

请使用以下步骤配置使用单一电源 fence 设备的节点，fence 设备名为 **apc**，使用 **fence_apc** fencing 代理。

1. 在该节点中添加 fence 方法，并为该 fence 方法提供名称。

```
ccs -h host --addmethod method node
```

例如：执行以下命令在集群节点 **node-01.example.com** 的配置文件中为节点 **node-01.example.com** 配置名为 **APC** 的 fence 方法：

```
ccs -h node01.example.com --addmethod APC node01.example.com
```

2. 为该方法添加 fence 事务。您必须为该节点指定要使用的 fence 设备，应用这个事务的节点，该方法的名称以及具体在这个节点中这个方法的所有选项：

```
ccs -h host --addfenceinst fencedevicename node method [options]
```

例如：请执行以下命令在集群节点 **node-01.example.com** 的配置文件中配置 fence 事务，该节点使用该 fence 设备中名为 **apc** 的 APC 电源切换端口 1 使用名为 **APC** 的方法 fence 集群节点 **node-01.example.com**：

```
ccs -h node01.example.com --addfenceinst apc node01.example.com APC
port=1
```

您需要为该集群的每个节点中添加 fence 方法。下面的命令使用名为 **APC** 方法为每个节点配置 fence 方法。使用该 fence 方法的设备指定 **apc** 作为设备名称，即之前使用 **--addfencedev** 选项指定的设备，如 第 5.5 节 “配置 Fence 设备” 所述。每一个节点都是使用唯一的 APC 切换电源端口号指定：**node-01.example.com** 的端口号为 **1**，**node-02.example.com** 的端口号为 **2**，**node-03.example.com** 的端口号为 **3**。

```
ccs -h node01.example.com --addmethod APC node01.example.com
ccs -h node01.example.com --addmethod APC node02.example.com
ccs -h node01.example.com --addmethod APC node03.example.com
ccs -h node01.example.com --addfenceinst apc node01.example.com APC port=1
ccs -h node01.example.com --addfenceinst apc node02.example.com APC port=2
ccs -h node01.example.com --addfenceinst apc node03.example.com APC port=3
```

例 5.2 “**cluster.conf** 添加使用电源的 Fence 方法后” 演示了在集群的每个节点中添加这些 fencing 方法和事务后的 **cluster.conf** 配置文件：

例 5.2. cluster.conf 添加使用电源的 Fence 方法后

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
```

```
            <fence>
               <method name="APC">
                 <device name="apc" port="2"/>
               </method>
            </fence>
        </clusternode>
        <clusternode name="node-03.example.com" nodeid="3">
            <fence>
               <method name="APC">
                 <device name="apc" port="3"/>
               </method>
            </fence>
        </clusternode>
    </clusternodes>
    <fencedevices>
            <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc" passwd="password_example"/>
    </fencedevices>
    <rm>
    </rm>
</cluster>
```

注：完成集群的所有组件配置后，需要在所有节点中同步该集群配置文件，如 第 5.15 节 “在集群节点中推广配置文件” 所述。

5.7.2. 为节点配置单一存储 Fence 设备

当使用非电源 fencing 方法（即 SAN/存储 fencing）fence 某个节点时，您必须为该 fence 设备配置 *unfencing*。这样可保证被 fence 的节点在重启该节点前不会重新启用。当您为某个节点配置 unfencing 时，您可以指定一个与对应 fence 设备成镜像的设备，该设备是您明确使用 **on** 或者 **enable** 为该节点配置的。

有关 unfencing 某个节点的详情请参考 **fence_node**(8) man page。

使用以下步骤配置使用单一存储 fence 设备的节点，该 fence 设备名为 **sanswitch1**，使用 **fence_sanbox2** fencing 代理。

1. 在该节点中添加 fence 方法，并为该 fence 方法提供名称。

   ```
   ccs -h host --addmethod method node
   ```

   例如：请执行以下命令在集群节点 **node-01.example.com** 的配置文件中为节点 **node-01.example.com** 配置名为 **SAN** 的 fence 方法：

   ```
   ccs -h node01.example.com --addmethod SAN  node01.example.com
   ```

2. 为该方法添加 fence 事务。您必须为该节点指定要使用的 fence 设备，应用这个事务的节点，该方法的名称以及具体在这个节点中这个方法的所有选项：

   ```
   ccs -h host --addfenceinst fencedevicename node method [options]
   ```

   例如：请执行以下命令在集群节点 **node-01.example.com** 的配置文件中配置 fence 事务，该

节点在名为 **sanswitch1** 的 fence 设备中使用 SAN 切换电源端口 11 fence 使用名为 **SAN** 方法的集群节点 **node-01.example.com**：

```
ccs -h node01.example.com --addfenceinst sanswitch1
node01.example.com SAN port=11
```

3. 请执行以下命令为这个节点中基于存储的 fence 设备配置 unfencing：

```
ccs -h host --addunfence fencedevicename node action=on|off
```

您需要为该集群中的每个节点添加 fence 方法。下面的命令为每个节点配置了名为 **SAN** 的 fence 方法。使用该 fence 方法的设备指定 **sanswitch** 为设备名称，它是之前使用 --addfencedev 选项配置的设备，如 第 5.5 节 “配置 Fence 设备” 所示。为每个节点配置一个唯一 SAN 物理端口号：**node-01.example.com** 的端口号为 **11**，**node-02.example.com** 的端口号为 **12**，**node-03.example.com** 的端口号为 **13**。

```
ccs -h node01.example.com --addmethod SAN node01.example.com
ccs -h node01.example.com --addmethod SAN node02.example.com
ccs -h node01.example.com --addmethod SAN node03.example.com
ccs -h node01.example.com --addfenceinst sanswitch1 node01.example.com SAN
port=11
ccs -h node01.example.com --addfenceinst sanswitch1 node02.example.com SAN
port=12
ccs -h node01.example.com --addfenceinst sanswitch1 node03.example.com SAN
port=13
ccs -h node01.example.com --addunfence sanswitch1 node01.example.com
port=11 action=on
ccs -h node01.example.com --addunfence sanswitch1 node02.example.com
port=12 action=on
ccs -h node01.example.com --addunfence sanswitch1 node03.example.com
port=13 action=on
```

例 5.3 “**cluster.conf** 添加基于存储的 Fence 方法后” 演示了在该集群的每个节点中添加 fencing 方法、fencing 事务以及 unfencing 后的配置文件。

例 5.3. cluster.conf 添加基于存储的 Fence 方法后

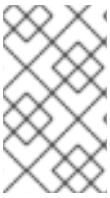
```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="SAN">
       <device name="sanswitch1" port="11"/>
            </method>
         </fence>
         <unfence>
            <device name="sanswitch1" port="11" action="on"/>
         </unfence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="SAN">
       <device name="sanswitch1" port="12"/>
```

```
            </method>
        </fence>
        <unfence>
            <device name="sanswitch1" port="12" action="on"/>
        </unfence>
    </clusternode>
    <clusternode name="node-03.example.com" nodeid="3">
        <fence>
           <method name="SAN">
      <device name="sanswitch1" port="13"/>
            </method>
        </fence>
        <unfence>
            <device name="sanswitch1" port="13" action="on"/>
        </unfence>
    </clusternode>
  </clusternodes>
  <fencedevices>
      <fencedevice agent="fence_sanbox2" ipaddr="san_ip_example"
login="login_example" name="sanswitch1" passwd="password_example"/>
  </fencedevices>
  <rm>
  </rm>
</cluster>
```

注：完成集群的所有组件配置后，需要在所有节点中同步该集群配置文件，如 第 5.15 节 “在集群节点中推广配置文件” 所述。

5.7.3. 配置备用 Fence 设备

您可以为一个节点定义多个 fencing 方法。如果使用第一个方法 fencing 失败，则系统会尝试使用第二个方法 fence 该节点，然后是其它您配置的方法。要为节点配置备用 fencing 方法，您需要为一个节点配置两个方法，并为每个方法配置一个 fence 事务。



注意

系统使用您所配置的 fencing 方法的顺序与其在配置文件中的顺序一致。使用 **ccs** 命令配置的第一个方法是首选 fencing 方法，您配置的第二个方法是备用 fencing 方法。要更改顺序，您可以从配置文件中删除首选 fencing 方法，然后再将其添加回配置文件中。

注：您可在任何时候执行以下命令输出当前为某个节点配置的 fence 方法和事务列表。如果您没有指定节点，这个命令将列出为所有节点当前配置的 fence 方法和事务。

```
ccs -h host --lsfenceinst [node]
```

使用以下步骤为某个节点配置首选 fencing 方法，该方法使用名为 **apc** 的 fence 设备，该设备使用 **fence_apc** fencing 代理，并配置使用名为 **sanswitch1** 的 fence 设备作为备用 fencing 设备，该设备使用 **fence_sanbox2** fencing 代理。因为 **sanswitch1** 设备是基于存储的 fencing 代理，所以您还需要为那个失败配置 unfencing。

1. 为该节点添加首选 fence 方法，并为该 fence 方法命名。
   -

```
ccs -h host --addmethod method node
```

例如：请执行以下命令在集群节点 **node-01.example.com** 的配置文件中为节点 **node-01.example.com** 将名为 **APC** 的 fence 方法配置为首选方法：

```
ccs -h node01.example.com --addmethod APC node01.example.com
```

2. 为首选方法添加 fence 事务。您必须指定要该节点要使用的 fence 设备，应用这个事务的节点，该方法的名称以及具体到这个节点该方法的所有选项：

```
ccs -h host --addfenceinst fencedevicename node method [options]
```

例如：请执行以下命令在集群节点 **node-01.example.com** 的配置文件中配置 fence 事务，该节点使用该 fence 设备中名为 **apc** 的 APC 电源切换端口 1 使用名为 **APC** 的方法 fence 集群节点 **node-01.example.com**：

```
ccs -h node01.example.com --addfenceinst apc node01.example.com APC
port=1
```

3. 为该节点添加备用 fence 方法，并提供该 fence 方法的名称。

```
ccs -h host --addmethod method node
```

例如：请执行以下命令为集群节点 **node-01.example.com** 配置文件中的节点 **node-01.example.com** 配置名为 **SAN** 的 fence 方法：

```
ccs -h node01.example.com --addmethod SAN  node01.example.com
```

4. 为该备用方法添加 fence 事务。您必须指定该节点要使用的 fence 设备，应用这个事务的节点，该方法的名称以及具体在这个节点中该方法要使用的所有选项：

```
ccs -h host --addfenceinst fencedevicename node method [options]
```

例如：请执行以下命令在集群节点 **node-01.example.com** 的配置文件中配置 fence 事务，该节点在名为 **sanswitch1** 的 fence 设备中使用 SAN 切换电源端口 11 fence 使用名为 **SAN** 方法的集群节点 **node-01.example.com**：

```
ccs -h node01.example.com --addfenceinst sanswitch1
node01.example.com SAN port=11
```

5. 因为 **sanswitch1** 设备是基于存储的设备，您必须为这个设备指定 unfencing。

```
ccs -h node01.example.com --addunfence sanswitch1 node01.example.com
port=11 action=on
```

您可以根据需要继续添加 fencing 方法。

这个过程为集群中的一个节点配置 fence 设备和备用 fence 设备。您还需要为该集群中的其它节点配置 fencing 。

例 5.4 “**cluster.conf** 添加备用 Fence 方法后” 演示了在该集群每个节点中，添加使用电源的主要 fencing 方和基于存储的备用 fencing 方法后的 **cluster.conf** 配置文件。

例 5.4. cluster.conf 添加备用 Fence 方法后

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
            <method name="SAN">
       <device name="sanswitch1" port="11"/>
             </method>
         </fence>
         <unfence>
             <device name="sanswitch1" port="11" action="on"/>
         </unfence
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
            <method name="SAN">
       <device name="sanswitch1" port="12"/>
             </method>
         </fence>
         <unfence>
             <device name="sanswitch1" port="12" action="on"/>
         </unfence
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
            <method name="APC">
              <device name="apc" port="3"/>
            </method>
            <method name="SAN">
       <device name="sanswitch1" port="13"/>
             </method>
         </fence>
         <unfence>
             <device name="sanswitch1" port="13" action="on"/>
         </unfence
     </clusternode>
   </clusternodes>
   <fencedevices>
        <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc" passwd="password_example"/>
        <fencedevice agent="fence_sanbox2" ipaddr="san_ip_example"
login="login_example" name="sanswitch1" passwd="password_example"/>
   </fencedevices>
   <rm>
   </rm>
```

```
</cluster>
```

注：完成集群的所有组件配置后，需要在所有节点中同步该集群配置文件，如 第 5.15 节 “在集群节点中推广配置文件” 所述。

注意

系统使用您配置的 fencing 方法的顺序与其在集群配置文件中的顺序一致。您配置的第一个方法就是首选 fencing 方法，您配置的第二个方法就是备用 fencing 方法。要更改顺序，您可以从配置文件中删除首选 fencing 方法，然后再添加回来。

5.7.4. 配置使用冗余电源的节点

如果您将集群配置为使用冗余电源供应您的节点，则您必须确定配置 fencing 以便在节点需要被 fence 时可完全关闭它们。如果您将每个电源供应都设定为独立的 fence 方法，则会分别 fence 每个电源供应；第二个电源供应可在 fence 了第一个电源供应时让该系统继续运行，同时根本不会 fence 该系统。要将系统配置为双电源供应，您必须配置 fence 设备以便可关闭两个电源并完全关闭系统。这要求您在一个 fencing 方法中配置两个事务，且在您配置的每个事务中，要在为设备配置 **action** 属性 **on** 之前配置 **action** 属性 **off**。

请按照本小节中的步骤为有双电源供应的节点配置 fencing。

1. 在您可以为使用冗余电源的节点配置 fencing 强，您必须为该集群将每个电源开关配置为 fence 设备。有关配置 fence 设备的详情请参考 第 5.5 节 “配置 Fence 设备”。

   请执行以下命令输出当前为您的集群配置的 fence 设备列表：

   ```
   ccs -h host --lsfencedev
   ```

2. 在该节点中添加 fence 方法，并为该 fence 方法提供名称。

   ```
   ccs -h host --addmethod method node
   ```

   例如：请执行以下命令在集群节点 **node-01.example.com** 的配置文件中为节点 **node-01.example.com** 配置名为 **APC-dual** 的 fence 方法：

   ```
   ccs -h node01.example.com --addmethod APC-dual node01.example.com
   ```

3. 为该 fence 方法的第一个电源添加 fence 事务。您必须指定该节点要使用的 fence 设备，应用这个事务的节点，该方法的名称以及具体在这个节点中可用于这个方法的所有选项。此时您可将 **action** 属性配置为 **on**。

   ```
   ccs -h host --addfenceinst fencedevicename node method [options]
   action=off
   ```

   例如：您可以执行以下命令在集群节点 **node-01.example.com** 的配置文件中配置 fence 事务，该事务在名为 **apc1** 的 fence 设备中使用 APC 切换电源端口 1 fence 集群几点 **node-01.example.com**，fence 方法名为 **APC-dual**，并将 **action** 属性设定为 **on**：

```
ccs -h node01.example.com --addfenceinst apc1 node01.example.com
APC-dual port=1 action=off
```

4. 为该 fence 方法的第二电源供应添加 fence 事务。您必须指定该节点要使用的 fence 设备，应用这个事务的节点，该方法的名称以及具体在这个节点中这个方法的所有选项。此时您也可为这个事务将 **action** 属性配置为 **off**：

```
ccs -h host --addfenceinst fencedevicename node method [options]
action=off
```

例如：请执行以下命令在集群节点 **node-01.example.com** 的配置文件中配置第二 fence 事务，该事务在名为 **apc2** 的 fence 设备中使用 APC 切换电源断口 1 fence 集群节点 **node-01.example.com**，该节点使用您为第一个事务所指定的同一方法，即 **APC-dual**，并将 **action** 属性设定为 **off**：

```
ccs -h node01.example.com --addfenceinst apc2 node01.example.com
APC-dual port=1 action=off
```

5. 此时，您可为给 fence 方法的第一电源供应添加另一个 fence 事务，将 **action** 属性配置为 **on**。您必须指定该节点要使用的 fence 设备，应用这个事务的节点，该方法的名称以及具体到这个节点这个方法的所有选项，并将 **action** 属性指定为 **on**：

```
ccs -h host --addfenceinst fencedevicename node method [options]
action=on
```

例如：请执行以下命令为集群节点 **node-01.example.com** 的配置文件配置 fence 事务，该事务使用名为 **apc1** 的 fence 设备的 APC 切换电源端口 1 fence 集群节点 **node-01.example.com**，使用的方法名为 **APC-dual**，并将 **action** 属性设定为 **on**：

```
ccs -h node01.example.com --addfenceinst apc1 node01.example.com
APC-dual port=1 action=on
```

6. 您可为给 fence 方法的第二电源供应添加另一个 fence 事务，将 **action** 属性指定为 **on**。您必须指定该节点要使用的 fence 设备，应用这个事务的节点，该方法的名称以及具体到这个节点这个方法的所有选项，并将 **action** 属性指定为 **on**：

```
ccs -h host --addfenceinst fencedevicename node method [options]
action=on
```

例如：请执行以下命令在集群节点 **node-01.example.com** 的配置文件中配置第二 fence 事务，该事务在名为 **apc2** 的 fence 设备中使用 APC 切换电源端口 1 fence 集群节点 **node-01.example.com**，该节点使用您为第一个事务所指定的同一方法，即 **APC-dual**，并将 **action** 属性设定为 **on**：

```
ccs -h node01.example.com --addfenceinst apc2 node01.example.com
APC-dual port=1 action=on
```

例 5.5 “**cluster.conf** 添加双电源 Fencing” 演示了在集群的每个端口中使用两个电源供应时添加 fencing 后的 **cluster.conf** 配置文件。

例 5.5. cluster.conf 添加双电源 Fencing

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC-dual">
              <device name="apc1" port="1"action="off"/>
              <device name="apc2" port="1"action="off"/>
              <device name="apc1" port="1"action="on"/>
              <device name="apc2" port="1"action="on"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC-dual">
              <device name="apc1" port="2"action="off"/>
              <device name="apc2" port="2"action="off"/>
              <device name="apc1" port="2"action="on"/>
              <device name="apc2" port="2"action="on"/>
            </method>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
            <method name="APC-dual">
              <device name="apc1" port="3"action="off"/>
              <device name="apc2" port="3"action="off"/>
              <device name="apc1" port="3"action="on"/>
              <device name="apc2" port="3"action="on"/>
            </method>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
       <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc1" passwd="password_example"/>
       <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc2" passwd="password_example"/>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

注：完成集群的所有组件配置后，需要在所有节点中同步该集群配置文件，如 第 5.15 节 “在集群节点中推广配置文件” 所述。

5.7.5. 删除 Fence 方法和 Fence 事务

请执行以下命令从您的配置文件中删除 fence 方法：

```
ccs -h host --rmmethod method node
```

例如：请执行以下命令在集群节点 **node01.example.com** 的集群配置文件中删除您为 **node01.example.com** 配置的名为 **APC** 的 fence 方法：

```
ccs -h node01.example.com  --rmmethod APC node01.example.com
```

请执行以下命令从 fence 方法中删除某个 fence 设备的所有 fence 事务：

```
ccs -h host --rmfenceinst fencedevicename node method
```

例如：请执行以下命令在集群节点 **node01.example.com** 的集群配置文件中从为 **node01.example.com** 配置的名为 **APC-dual** 的方法中删除 fence 设备名为 **apc1** 的所有事务：

```
ccs -h node01.example.com --rmfenceinst apc1 node01.example.com APC-dual
```

5.8. 配置故障切换域

故障切换域是节点集群的一个命名子集，可在节点失败事件中运行集群服务。故障切换域有以下特征：

- 无限制 — 可让您指定您喜欢的子集成员，但分配给这个域的集群服务可在任意成员中运行。
- 限制的 — 可让您限制可运行具体集群服务的成员。如果在限制的故障切换域中没有任何成员可用，则无法启动该集群服务（手动或者使用集群软件都不行）。
- 无序的 — 当为某个无序的故障切域分配集群服务时，运行该集群服务的域的成员是从可用故障切换域成员中随即挑选的。
- 有序的 — 可让您在故障切换域的成员间指定顺序。该列表顶端的成员是首选成员，接下来是列表中的第二个成员，依此类推。
- 返回 — 可让您指定是否让故障切换域中的服务返回节点失败前运行该服务的节点中。配置这个特性对节点会反复失败的环境很有用，且该节点是有序故障切换域的一部分。在那个环境中，如果某个节点是故障切换域中的首选节点，那么某个服务就可能不断在首选节点和另一个节点间进行故障切换和返回，从而影响服务器性能。

注意

只有在配置了有序故障切换时方可应用返回属性。

注意

更改故障切换域配置对当前运行中的服务没有影响。

注意

操作*不*需要故障切换域。

默认情况下故障切换域是无限制且无序的。

在有几个成员的集群中，使用限制的故障切换域可尽量减少配置集群运行集群服务（比如 **httpd**）的工作，这些配置要求您在运行该集群服务的所有成员中设定完全相同的配置。与其设定整个集群运行该集群服务，您可以只在与该集群服务关联的限制故障切换域成员中进行设定。

注意

要配置首选成员，您可以创建只有一个集群成员的无限制故障切换域。这样做就让集群服务主要在那个集群成员（首选成员）中运行，但允许将该集群服务故障切换到任意其它成员中。

请执行以下步骤配置故障切换域：

1. 请执行以下命令添加故障切换域：

```
ccs -h host --addfailoverdomain name [restricted] [ordered]
[nofailback]
```

注意

该名称应可显示出与您集群中使用的其它名称在使用目的上有区别。

例如：以下命令在无限制、有序且允许返回的 **node-01.example.com** 中，配置名为 **example_pri** 的故障切换域：

```
ccs -h node-01.example.com --addfailoverdomain example_pri ordered
```

2. 请执行以下命令在故障切换域中添加一个节点：

```
ccs -h host --addfailoverdomainnode failoverdomain node priority
```

例如：请执行以下命令在 **node-01.example.com** 的配置文件中配置故障切换域 **example_pri**，这样它就包括优先权为 1 的 **node-01.example.com**，优先权为 2 的 **node-02.example.com** 和优先权为 3 的 **node-03.example.com**：

```
ccs -h node-01.example.com --addfailoverdomainnode example_pri node-
01.example.com 1
ccs -h node-01.example.com --addfailoverdomainnode example_pri node-
02.example.com 2
ccs -h node-01.example.com --addfailoverdomainnode example_pri node-
03.example.com 3
```

您可使用以下命令列出在集群中配置的所有故障切换域和故障切换域节点：

```
ccs -h host --lsfailoverdomain
```

请执行以下命令删除故障切换域：

```
ccs -h host --rmfailoverdomain name
```

请执行以下命令从故障切换域中删除节点：

```
ccs -h host --rmfailoverdomainnode failoverdomain node
```

注：完成集群的所有组件配置后，需要在所有节点中同步该集群配置文件，如 第 5.15 节 “在集群节点中推广配置文件” 所述。

5.9. 配置全局集群资源

您可以配置两类资源：

- 全局 — 可在集群的任意服务中使用的资源。
- 具体服务 — 只可用于一个服务的资源。

请执行以下命令查看当前在该集群中配置的资源和服务列表：

```
ccs -h host --lsservices
```

请执行以下命令添加全局集群资源。您可在配置该资源时添加某个具体服务的本地资源，如 第 5.10 节 “在集群中添加集群服务” 所述。

```
ccs -h host --addresource resourcetype [resource options]
```

例如：下面的命令在 **node01.example.com** 的集群配置文件中添加了一个全局文件系统资源。该资源的名称为 **web_fs**，该文件系统设备为 **/dev/sdd2**，挂载点为 **/var/www**，类型为 **ext3**。

```
ccs -h node01.example.com --addresource fs name=web_fs device=/dev/sdd2
mountpoint=/var/www fstype=ext3
```

有关可用资源类型和资源选项的详情请参考 附录 B, *HA 资源参数*。

请执行以下命令删除全局资源：

```
ccs -h host --rmresource resourcetype [resource options]
```

如果您要修改现有全局资源参数，您可以删除该资源然后再重新配置。

注：完成集群的所有组件配置后，需要在所有节点中同步该集群配置文件，如 第 5.15 节 “在集群节点中推广配置文件” 所述。

5.10. 在集群中添加集群服务

请执行以下步骤在集群中配置集群服务：

1. 请使用以下命令在集群中添加服务：

   ```
   ccs -h host --addservice servicename [service options]
   ```

   注意

   为该服务使用可将其与集群中其他服务明确区分开来的说明性名称。

   请在向集群配置中添加服务是配置以下属性：

   - **autostart** — 指定是否在集群启动时自动启动该服务。使用“1”启用，“0”禁用，默认为启用。
   - **domain** — 指定故障切换域（如果需要）。

- **exclusive** — 指定服务只能在没有其他服务运行的节点中运行的策略。

- **recovery** — 为该服务指定恢复策略。选项有 relocate、restart、disable 或者 restart-disable 该服务。restart 恢复策略表示应在另一节点中重新定位该服务前尝试重启失败的服务。relocate 策略表示系统应在不同节点中重启该服务。disable 策略表示如果任意组件失败，则系统应禁用该资源组。restart-disable 策略表示系统应在服务失败的地方尝试重启该服务，但如果重启失败，则会禁用该服务，而不是移动到集群的另一台主机中。

  如果您选择 **Restart** 或者 **Restart-Disable** 作为该服务的恢复策略，您可以指定重新定位或者禁用该服务前最多重启失败的次数，您还可以在多少秒后不再重启。

例如：请执行以下命令在集群节点 **node-01.example.com** 的配置文件中添加名为 **example_apache** 的服务，该服务使用故障切换域 **example_pri**，恢复策略为 **relocate**：

```
ccs -h node-01.example.com --addservice example_apache
domain=example_pri recovery=relocate
```

为集群配置服务时，您可能会发现您集群的可用服务列表以及每个服务的可用选项列表很有帮助。有关使用 **ccs** 命令输出可用服务及其选项列表的详情请参考 第 5.11 节 “列出可用集群服务”。

2. 请使用以下命令在服务中添加资源：

```
ccs -h host --addsubservice servicename subservice [service options]
```

请根据您想要使用的资源类型使用全局或者具体服务资源传播服务。请在添加资源时使用 **ccs** 命令的 **--addsubservice** 选项添加全局资源。例如：请执行以下命令在 **node-01.example.com** 集群配置文件中为名为 **example_apache** 的服务添加名为 **web_fs** 的全局文件系统资源：

```
ccs -h node01.example.com --addsubservice example_apache fs
ref=web_fs
```

要在服务中添加具体服务资源，您需要指定所有服务选项。例如：如果您之前没有将 **web_fs** 定义为全局服务，则您可以使用以下命令将其添加为具体服务资源：

```
ccs -h node01.example.com --addsubservice example_apache fs
name=web_fs device=/dev/sdd2 mountpoint=/var/www fstype=ext3
```

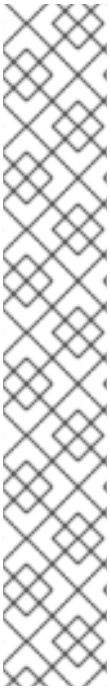
3. 要在该服务中添加子服务，您还可以使用 **ccs** 命令的 **--addsubservice** 选项指定服务选项。

如果您需要在相依性树状结构中添加服务，请使用冒号（":"）分隔元素，并使用括号区分同一类型的子服务。下面的示例添加了第三个 **nfsclient** 服务作为 **nfsclient** 服务的子服务，它本身是 **nfsclient** 服务的子服务，而后者又是 **service_a** 服务的子服务：

```
ccs -h node01.example.com --addsubservice service_a
nfsclient[1]:nfsclient[2]:nfsclient
```

注意

如果您要添加 Samba 服务资源，请直接在该服务中添加，*不要*将其作为另一个资源的子资源使用。



注意

要确认在集群服务中使用的 IP 服务资源，您可以在集群节点中使用 **/sbin/ip addr show** 命令而不是弃用的 **ifconfig** 命令。以下显示了在运行集群服务的节点中运行 **/sbin/ip addr show** 的输出结果：

```
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1356 qdisc pfifo_fast
qlen 1000
    link/ether 00:05:5d:9a:d8:91 brd ff:ff:ff:ff:ff:ff
    inet 10.11.4.31/22 brd 10.11.7.255 scope global eth0
    inet6 fe80::205:5dff:fe9a:d891/64 scope link
    inet 10.11.4.240/22 scope global secondary eth0
       valid_lft forever preferred_lft forever
```

请执行以下命令删除服务及其所有子服务：

```
ccs -h host --rmservice servicename
```

请执行以下命令删除子服务：

```
ccs -h host --rmsubservice servicename subservice [service options]
```

注：完成集群的所有组件配置后，需要在所有节点中同步该集群配置文件，如 第 5.15 节 “在集群节点中推广配置文件” 所述。

5.11. 列出可用集群服务

您可以使用 **ccs** 命令输出您集群现在可使用的服务列表。还可以使用 **ccs** 命令输出可为具体服务类型指定的选项列表。

请使用以下命令输出您集群现在可用的集群服务列表：

```
ccs -h host --lsserviceopts
```

例如：以下命令列出集群节点 **node1** 中可用的集群服务，演示示例输出结果：

```
[root@ask-03 ~]# ccs -h node1 --lsserviceopts
service - Defines a service (resource group).
ASEHAagent - Sybase ASE Failover Instance
SAPDatabase - SAP database resource agent
SAPInstance - SAP instance resource agent
apache - Defines an Apache web server
clusterfs - Defines a cluster file system mount.
fs - Defines a file system mount.
ip - This is an IP address.
lvm - LVM Failover script
mysql - Defines a MySQL database server
named - Defines an instance of named server
```

```
netfs - Defines an NFS/CIFS file system mount.
nfsclient - Defines an NFS client.
nfsexport - This defines an NFS export.
nfsserver - This defines an NFS server resource.
openldap - Defines an Open LDAP server
oracledb - Oracle 10g Failover Instance
orainstance - Oracle 10g Failover Instance
oralistener - Oracle 10g Listener Instance
postgres-8 - Defines a PostgreSQL server
samba - Dynamic smbd/nmbd resource agent
script - LSB-compliant init script as a clustered resource.
tomcat-6 - Defines a Tomcat server
vm - Defines a Virtual Machine
action - Overrides resource action timings for a resource instance.
```

请使用以下命令输出可为具体服务类型指定的选项列表：

```
ccs -h host --lsserviceopts service_type
```

例如：下面的命令列出 **vm** 服务的服务选项。

```
[root@ask-03 ~]# ccs -f node1 --lsserviceopts vm
vm - Defines a Virtual Machine
  Required Options:
    name: Name
  Optional Options:
    domain: Cluster failover Domain
    autostart: Automatic start after quorum formation
    exclusive: Exclusive resource group
    recovery: Failure recovery policy
    migration_mapping: memberhost:targethost,memberhost:targethost ..
    use_virsh: If set to 1, vm.sh will use the virsh command to manage
virtual machines instead of xm. This is required when using non-Xen
virtual machines (e.g. qemu / KVM).
    xmlfile: Full path to libvirt XML file describing the domain.
    migrate: Migration type (live or pause, default = live).
    path: Path to virtual machine configuration files.
    snapshot: Path to the snapshot directory where the virtual machine
image will be stored.
    depend: Top-level service this depends on, in service:name format.
    depend_mode: Service dependency mode (soft or hard).
    max_restarts: Maximum restarts for this service.
    restart_expire_time: Restart expiration time; amount of time before a
restart is forgotten.
    status_program: Additional status check program
    hypervisor: Hypervisor
    hypervisor_uri: Hypervisor URI (normally automatic).
    migration_uri: Migration URI (normally automatic).
    __independent_subtree: Treat this and all children as an independent
subtree.
    __enforce_timeouts: Consider a timeout for operations as fatal.
    __max_failures: Maximum number of failures before returning a failure
to a status check.
    __failure_expire_time: Amount of time before a failure is forgotten.
    __max_restarts: Maximum number restarts for an independent subtree
```

```
before giving up.
    __restart_expire_time: Amount of time before a failure is forgotten
for an independent subtree.
```

5.12. 虚拟机资源

配置虚拟机资源与配置其他集群资源不同。特别是它们不是根据服务定义分组的。从红帽企业版 Linux 6.2 开始，当在集群中使用 **ccs** 命令配置虚拟机时，可以使用 **--addvm**（而不是 **addservice** 选项）。这样就可保证在集群配置文件 **rm** 配置节点下直接定义 **vm** 资源。

虚拟机资源至少需要一个 **name** 和一个 **path** 属性。**name** 属性应与 **libvirt** 域的名称相符，而 **path** 属性应指定保存共享虚拟机定义的目录。

注意

集群配置文件中的 **path** 属性是路径说明，或者目录名，而不是到独立文件的路径。

如果在名为 **/mnt/vm_defs** 的共享目录中保存虚拟机定义，则下面的命令则定义名为 **guest1** 的虚拟机：

```
# ccs -h node1.example.com --addvm guest1 path=/mnt/vm_defs
```

运行这个命令在 **cluster.conf** 配置文件的 **rm** 配置节点中添加以下行：

```
<vm name="guest1" path="/mnt/vm_defs"/>
```

5.13. 配置仲裁磁盘

注意

Quorum-disk 参数以及探测法根据网站环境以及特殊需要有所不同。要了解如何使用 Quorum-disk 参数和探测法，请参考 qdisk(5) man page。如果您在理解和使用仲裁磁盘方面需要帮助，请联络授权的红帽支持代表。

请使用以下命令将您的系统配置为使用仲裁磁盘：

```
ccs -h host --setquorumd [quorumd options]
```

注：这个命令会将您使用 **--setquorumd** 选项设定的其他所有属性重新设置为恢复到其默认值，如 第 5.1.5 节 “覆盖之前设置的命令” 所示。

表 5.1 “仲裁磁盘选项” 总结了您可能需要设置的仲裁磁盘选项的含义。有关仲裁磁盘参数的完整列表请参考 **/usr/share/cluster/cluster.rng** 中的集群方案以及 **/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html** 中的注释方案。

表 5.1. 仲裁磁盘选项

参数	描述

参数	描述
「间隔」	读取/写入循环的频率，单位为秒。
「投票」	仲裁守护进程告知 **cman** 它有足够分数时的票数。
「tko」	宣布节点死亡时节点必须错过的循环数。
「最低分数」	将节点视为 "alive" 的最小分数。如果省略或者设为 0，则使用默认功能 **floor((*n*+1)/2)**，其中 *n* 为探测法分数之和。**Minimum Score** 值永远不能超过探测法分数之和，否则将无法使用该仲裁磁盘。
「设备」	制裁守护进程使用的存储设备。在所有节点中该设备必须相同。
「标签」	指定由 **mkqdisk** 工具创建的制裁磁盘标签。如果这个字段中包含一个条目，则该标签将覆盖 **Device** 字段。如果这个字段已经被使用，则仲裁守护进程会在每个找到的块设备中读取 **/proc/partitions** 并检查 qdisk 签名，根据指定的标签对比该标签。这在节点间使用不同仲裁设备名称配置时有用。

请使用以下命令为仲裁磁盘配置探测法：

```
ccs -h host --addheuristic [heuristic options]
```

表 5.2 “仲裁磁盘探测法” 总结了您可能需要设置的仲裁磁盘探测法的含义。

表 5.2. 仲裁磁盘探测法

参数	描述
「程序」	使用到程序的路径决定这个试探是否可用。它是 **/bin/sh -c** 可执行的任意程序。返回值为 0 表示成功；其他则表示失败。这是必填项。
「间隔」	调用探测法的频率（单位为秒）。每个探测法间的默认间隔为 2 秒。
「分数」	这个探测法的加权。决定探测法分数时请小心。每个探测法的默认分数为 1。
「tko」	在宣布这个探测法不可用前连续失败的次数。

您可执行以下命令查看系统配置的仲裁磁盘选项和探测法：

```
ccs -h host --lsquorum
```

您可执行以下命令删除探测法选项指定的探测法：

```
ccs -h host rmheuristic [heuristic options]
```

注：完成集群的所有组件配置后，需要在所有节点中同步该集群配置文件，如 第 5.15 节 “在集群节点中推广配置文件” 所述。

注意

同步和激活推广，并激活更新的集群配置文件。但要让仲裁磁盘可操作，您必须重启该集群（请参考 第 6.2 节 “启动和停止集群”），这样方可保证在每个节点中都重启 **qdiskd** 守护进程。

5.14. 其他集群配置

本小节描述了如何使用 **ccs** 命令配置以下内容：

- 第 5.14.1 节 “集群配置版本”
- 第 5.14.2 节 “多播配置”
- 第 5.14.3 节 “配置双节点集群”
- 第 5.14.4 节 “日志”
- 第 5.14.5 节 “配置冗余环协议”

您还可以使用 **ccs** 命令设定高级集群配置参数，其中包括 **totem** 选项、**dlm** 选项、**rm** 选项和 **cman** 选项。有关设定这些参数的详情请参考 **ccs**(8) man page 以及 **/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html** 中注释的集群配置文件方案。

请执行以下命令查看为集群配置的其它集群属性：

```
ccs -h host --lsmisc
```

5.14.1. 集群配置版本

集群配置版本包括集群配置版本值。默认在创建集群配置文件时将配置版本值设定为 **1**，并在每次您修改集群配置时自动增加。但如果您将其设定为其它数值，您可以使用以下命令指定该数值：

```
ccs -h host --setversion n
```

您可使用以下命令在获得现有集群配置文件的当前配置版本值：

```
ccs -h host --getversion
```

请执行以下命令将集群中每个节点的集群配置文件中的当前配置版本值加 1：

```
ccs -h host --incversion
```

5.14.2. 多播配置

如果您没有在集群配置文件中指定多播地址，红帽高可用性附加组件软件可根据集群 ID 创建一个。它可生成地址的后 16 位数字，并根据所使用的 IP 协议（IPV4 或者 IPV6）将其附加到该地址中：

- 对于 IPv4 — 该地址格式为 239.192 加上红帽高可用性附加组件软件生成的后 16 字节。
- 对于 IPv6 — 该地址格式为 FF15:: 加上红帽高可用性附加组件软件生成的后 16 字节。

注意

集群 ID 是 **cman** 为每个集群生成的唯一识别符。请在集群节点中运行 **cman_tool status** 命令查看集群 ID。

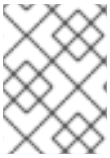
您可使用以下命令在集群配置文件中手动指定多播地址：

```
ccs -h host --setmulticast multicastaddress
```

注：这个命令会将您使用 **--setmulticast** 选项设定的其他所有属性重新设置为恢复到其默认值，如 第 5.1.5 节 “覆盖之前设置的命令” 所示。

如果您指定了多播地址，您应该使用 **cman** 使用的 239.192.x.x 系列（IPv6 使用 FF15::）。使用该范围以外的多播地址将导致不可预测的结果。例如：使用 224.0.0.x（即“网络中的所有主机”）可能无法正确路由，或者在有些硬件中根本无法路由。

如果您指定或修改多播地址，则必须重启该集群以便其生效。有关使用 **ccs** 命令启动和停止集群的详情请参考 第 6.2 节 “启动和停止集群”。



注意

如果您指定了多播地址，请确定检查集群数据包通过的路由器配置。有些路由器可能需要较长时间了解这些地址，这样会严重影响集群性能。

要删除配置文件中的多播地址，请使用 **ccs** 的 **--setmulticast** 选项，但不要指定多播地址：

```
ccs -h host --setmulticast
```

5.14.3. 配置双节点集群

如果您要配置双节点集群，您可执行以下命令允许单一节点维护仲裁（例如：如果一个节点失败）：

```
ccs -h host --setcman two_node=1 expected_votes=1
```

注：这个命令会将您使用 **--setcman** 选项设定的其他所有属性重新设置为恢复到其默认值，如 第 5.1.5 节 “覆盖之前设置的命令” 所示。

使用 **ccs --setcman** 命令添加、删除或修改 **two_node** 选项时，必须重启集群方可使更改生效。有关使用 **ccs** 命令启用和停止集群的详情请参考 第 6.2 节 “启动和停止集群”。

5.14.4. 日志

您可以为集群中的所有守护进程启用 debugging，或者您可以为具体集群进程启用日志。

要为所有守护进程启用 debugging，请执行以下命令。默认情况下会将日志指向 **/var/log/cluster/*daemon*.log** 文件。

```
ccs -h host --setlogging [logging options]
```

例如：下面的命令为所有守护进程启用 debugging。

```
# ccs -h node1.example.com --setlogging debug=on
```

注：这个命令会将您使用 **--setlogging** 选项设定的其他所有属性重新设置为恢复到其默认值，如 第 5.1.5 节 “覆盖之前设置的命令” 所示。

要为独立集群进程启用 debugging，请执行以下命令。个体守护进程日志配置可覆盖全局设置。

```
ccs -h host --addlogging [logging daemon options]
```

例如：下面的命令启用 **corosync** 和 **fenced** 守护进程的 debugging。

```
# ccs -h node1.example.com --addlogging name=corosync debug=on
# ccs -h node1.example.com --addlogging name=fenced debug=on
```

请使用以下命令删除独立守护进程的日志设置：

```
ccs -h host --rmlogging name=clusterprocess
```

例如：下面的命令删除 **fenced** 守护进程的具体守护进程日志设置。

```
ccs -h host --rmlogging name=fenced
```

对于您可以启用日志的日志守护进程列表以及您可以同时配置全局和按守护进程记录日志的附加日志选项，请参考 **cluster.conf**(5) man page。

注：完成集群的所有组件配置后，需要在所有节点中同步该集群配置文件，如 第 5.15 节 “在集群节点中推广配置文件” 所述。

5.14.5. 配置冗余环协议

从红帽企业版 Linux 6.4 开始，红帽高可用附加组件支持冗余环协议配置。当使用冗余环协议时，您需要考虑以下事项，如 第 7.6 节 “配置冗余环协议” 所述。

要为冗余环协议指定辅网络接口，请使用 **ccs** 命令的 **--addalt** 选项添加该节点的备用名称：

```
ccs -h host --addalt node_name alt_name
```

例如：以下命令为集群节点 **clusternet-node1-eth1** 配置备用名称 **clusternet-node1-eth2**：

```
# ccs -h clusternet-node1-eth1 --addalt clusternet-node1-eth1 clusternet-
node1-eth2
```

另外，您可以为第二个环手动指定多播地址、端口和 TTL。如果您为第二个环指定多播地址，要么使用备用多播地址，要么备用端口必须与第一个环的多播地址不同。如果您要指定备用端口，则第一个环和第二个环的端口号之差必须大于 2，因为该系统本身使用端口和端口-1 执行操作。如果您没有指定备用多播地址，该系统会自动为第二个环使用不同的多播地址。

要为第二个环指定备用多播地址、端口或者 TTL，可使用 **ccs** 命令的 **--setaltmulticast** 选项：

```
ccs -h host --setaltmulticast [alt_multicast_address]
[alt_multicast_options].
```

例如：以下命令会为 **cluster.conf** 文件中定义的集群在节点 **clusternet-node1-eth1** 中设定备用多播地址 239.192.99.88、端口 888 和 TTL 3：

```
ccs -h clusternet-node1-eth1 --setaltmulticast 239.192.99.88 port=888
ttl=3
```

要删除备用多播地址，请使用 **ccs** 命令的 **--setaltmulticast** 选项，但不要指定多播地址。注：执行这个命令会重置您使用 **--setaltmulticast** 选项设定其他所有选项，并将其恢复到默认值，如 第 5.1.5 节 “覆盖之前设置的命令” 所述。

您完成配置集群的所有组件后，需要在所有节点中同步该集群配置文件，如 第 5.15 节 “在集群节点中推广配置文件” 所述。

5.15. 在集群节点中推广配置文件

您在集群节点之一创建或者编辑集群配置文件后，需要将同一文件传播到所有集群节点并激活该配置。

使用以下命令传播并激活集群配置文件：

```
ccs -h host --sync --activate
```

请执行以下命令确定主机集群配置文件中指定的所有节点有相同的集群配置文件：

```
ccs -h host --checkconf
```

如果您在本地节点创建或者编辑了配置文件，请执行以下命令将其发送到该集群的一个节点中：

```
ccs -f file -h host --setconf
```

请执行以下命令验证该本地文件中指定的所有节点有相同的集群配置文件：

```
ccs -f file --checkconf
```

第 6 章 使用 CCS 管理 RED HAT 高可用性附加组件

本章论述了使用 **ccs** 工具管理 Red Hat 高可用性附加组件的各种管理任务，在红帽企业版 Linux 6.1 以及之后的版本中支持这个工具。本章由以下小节组成：

- 第 6.1 节 “管理集群节点”
- 第 6.2 节 “启动和停止集群”
- 第 6.3 节 “诊断并修正集群中的问题”

6.1. 管理集群节点

本小节记录了如何使用 **ccs** 命令执行以下节点管理功能：

- 第 6.1.1 节 “使节点离开或者加入集群”
- 第 6.1.2 节 “在运行的集群中添加成员”

6.1.1. 使节点离开或者加入集群

您可以使用 **ccs** 命令，通过停止在那个节点中的集群服务让节点离开集群。让节点离开集群不会从那个节点中删除集群配置信息。让节点离开集群可防止在重启时将其自动加入该集群。

请执行以下命令让节点离开集群，指定 **-h** 选项可停止该节点中的集群服务：

```
ccs -h host --stop
```

您停止节点中的集群服务时，会故障切换所有在该节点中运行的服务。

要从集群配置文件中删除整个节点，请使用 **ccs** 命令的 **--rmnode** 选项，如 第 5.4 节 “创建集群” 所述。

请执行以下命令让节点重新加入集群，可指定 **-h** 选项启动启动该节点中的集群服务：

```
ccs -h host --start
```

6.1.2. 在运行的集群中添加成员

要在运行的集群中添加成员，可如 第 5.4 节 “创建集群” 所述在集群中添加节点。更新配置文件后，请将其推广到该集群的所有节点中，并确定激活了新的集群配置文件，如 第 5.15 节 “在集群节点中推广配置文件” 所述。

6.2. 启动和停止集群

您可以使用 **ccs** 命令停止集群，并使用以下命令停止集群中所有节点中的集群服务：

```
ccs -h host --stopall
```

您可以使用 **ccs** 命令启动集群，并使用以下命令启动集群中所有节点中的集群服务：

```
ccs -h host --startall
```

6.3. 诊断并修正集群中的问题

有关诊断并修正集群中问题的详情请参考 第 9 章 *诊断并修正集群中的问题*。这里有一些您可以使用 **ccs** 命令执行的简单检查。

请执行以下命令验证主机集群配置文件中的所有节点是否有相同的集群配置文件：

```
ccs -h host --checkconf
```

如果是在本地节点中创建或者编辑配置文件，则可以使用以下命令验证在本地文件中指定的所有节点是否有相同的集群配置文件：

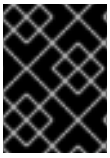
```
ccs -f file --checkconf
```

第 7 章 使用命令行工具配置红帽高可用附加组件

本章论述了如何通过直接编辑集群配置文件（**/etc/cluster/cluster.conf**）以及使用命令行工具配置红帽高可用性附加组件软件。本章分节提供了有关构建配置文件的步骤，从本章提供的简单文件开始。另外，您可以使用这里提供的简单文件，从 **cluster.conf** man page 复制配置文件框架。但这样做就无法与本章之后所提供步骤的信息对应。还有其它方法可创建并配置集群文件。本章分节提供有关构建配置文件的步骤。另外请记住这只是编写适合您集群所需配置文件的开始。

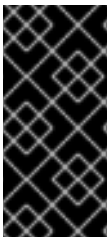
本章由以下节组成：

- 第 7.1 节 “配置任务”
- 第 7.2 节 “生成配置基本集群配置文件”
- 第 7.3 节 “配置 Fencing”
- 第 7.4 节 “配置故障切换域”
- 第 7.5 节 “配置 HA 服务”
- 第 7.7 节 “配置 Debug 选项”
- 第 7.6 节 “配置冗余环协议”
- 第 7.8 节 “验证配置”



重要

请确定您部署的高可用性附加组件满足您的需要并可被支持。部署前请咨询授权红帽代表确认您的配置。另外预留足够的时间测试配置的失败模式。



重要

本章通常使用 **cluster.conf** 元素和属性作为参考。有关 **cluster.conf** 元素和属性的完整列表，请参考 **/usr/share/cluster/cluster.rng** 中的集群方案，注释的方案位于 **/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html**（例如：**/usr/share/doc/cman-3.0.12/cluster_conf.html**）。

重要

本章的某些步骤调用 **cman_tool version -r** 命令在整个集群中推广集群配置。使用该命令需要运行 **ricci**。使用 **ricci** 要求您首次从某台具体机器与 **ricci** 互动时输入密码。有关 **ricci** 服务的详情请参考 第 2.13 节 “**ricci** 注意事项”。

注意

本章中的步骤可能包括某些命令行工具的具体命令，如 附录 E, *命令行工具小结* 所示。有关所有命令和变量的详情请参考每个命令行工具的 man page。

7.1. 配置任务

使用命令行工具配置红帽高可用性附加组件步骤如下：

1. 创建一个集群。请参考 第 7.2 节 “生成配置基本集群配置文件”。
2. 配置 fencing。请参考 第 7.3 节 “配置 Fencing”。
3. 配置故障切换域。请参考 第 7.4 节 “配置故障切换域”。
4. 配置 HA 服务。请参考 第 7.5 节 “配置 HA 服务”。
5. 验证配置。请参考 第 7.8 节 “验证配置”。

7.2. 生成配置基本集群配置文件

如果安装了集群硬件、红帽企业版 Linux 和高可用性附加组件软件，您就可以创建一个集群配置文件（**/etc/cluster/cluster.conf**）并开始运行高可用性附加组件。本小节论述了如何创建没有 fencing、故障切换以及 HA 服务的配置文件框架，这只是个开始。后续的小节将论述如何在配置文件中配置那些部分。

重要

这只是创建集群配置文件的临时步骤，所生成的文件不包含任何 fencing，也不会将其视为支持的配置。

以下步骤描述了如何创建并配置集群配置文件框架。最终，您集群的配置文件在节点数、fencing 类型、HA 服务的类型及数目以及其它具体位置要求方面会有所不同。

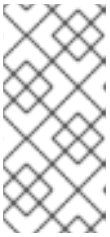
1. 在集群中的任意节点使用 例 7.1 “**cluster.conf** 示例：基本配置” 中的示例模板创建 **/etc/cluster/cluster.conf**。
2. **（可选）**，如果您要配置有两个节点的集群，您可以在配置文件中添加以下行允许单节点以便维持法定数（例如：如果一个节点失败）：

   ```
   <cman two_node="1" expected_votes="1"/>
   ```

   您在 **cluster.conf** 文件中添加或删除 **two_node** 选项时，必须重启该集群以便在更新配置时使更改生效。有关更新集群配置的详情请参考 第 8.4 节 “更新配置”。指定 **two_node** 选项的示例请参考 例 7.2 “**cluster.conf** 示例：基本双节点配置”。
3. 使用 **cluster** 属性：**name** 和 **config_version** 指定集群名称以及配置版本号（请参考 例 7.1 “**cluster.conf** 示例：基本配置” 或者 例 7.2 “**cluster.conf** 示例：基本双节点配置”）。
4. 在 **clusternodes** 部分，请使用 **clusternode** 属性：**name** 和 **nodeid** 为每个节点指定节点名称和节点 ID。
5. 保存 **/etc/cluster/cluster.conf**。
6. 根据集群方案（**cluster.rng**）通过运行 **ccs_config_validate** 命令验证该文件。例如：

   ```
   [root@example-01 ~]# ccs_config_validate
   Configuration validates
   ```

7. 将配置文件推广到每个集群节点的 **/etc/cluster/**。例如：您可以使用 **scp** 命令将该文件推广到其他集群节点中。



注意

在首次创建集群时需要使用这个方法推广集群配置文件。安装并运行集群后，可使用 **cman_tool version -r** 传推广群配置文件。可以使用 **scp** 推广更新的配置文件，但使用 **scp** 命令时必须在所有节点中停止集群软件。另外，如果您使用 **scp** 推广更新的配置文件，则应该运行 **ccs_config_validate**。

注意

当在同一配置文件中有其他元素和属性时（例如：**fence** 和 **fencedevices**），就不需要现在推广它们。本章后面的步骤提供有关指定其他元素和属性的信息。

8. 启动集群。在每个集群节点中运行以下命令：

   service cman start

   例如：

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                             [  OK
]
   Global setup...                                         [  OK
]
   Loading kernel modules...                               [  OK
]
   Mounting configfs...                                    [  OK
]
   Starting cman...                                        [  OK
]
   Waiting for quorum...                                   [  OK
]
   Starting fenced...                                      [  OK
]
   Starting dlm_controld...                                [  OK
]
   Starting gfs_controld...                                [  OK
]
   Unfencing self...                                       [  OK
]
   Joining fence domain...                                 [  OK
]
```

9. 在任意集群节点中运行 **cman_tool nodes**，确认那些节点作为集群的成员运行（在状态列 "Sts" 中被标记为 "M"）。例如：

```
[root@example-01 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M    548   2010-09-28 10:52:21  node-01.example.com
   2   M    548   2010-09-28 10:52:21  node-02.example.com
   3   M    544   2010-09-28 10:52:21  node-03.example.com
```

10. 如果集群正在运行，请执行 第 7.3 节 “配置 Fencing”。

基本配置示例

例 7.1 “**cluster.conf** 示例：基本配置” 和 例 7.2 “**cluster.conf** 示例：基本双节点配置”（双节点集群）提供最基本的集群配置示例。本章的后续步骤将提供有关配置 fencing 和 HA 服务的信息。

例 7.1. cluster.conf 示例：基本配置

```
<cluster name="mycluster" config_version="2">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

例 7.2. cluster.conf 示例：基本双节点配置

```
<cluster name="mycluster" config_version="2">
   <cman two_node="1" expected_votes="1"/>
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

双节点集群中 totem 的 consensus 值

您创建双节点集群且不想之后为该集群添加附加节点时，您应该忽略 **cluster.conf** 文件 **totem** 标签的 **consensus** 值，以便根据以下规则自动计算 **consensus** 值：

- 如果有两个或者少于两个节点，**consensus** 值将在 200 毫秒到 2000 毫秒之间（token * 0.2）。
- 如果有三个或者三个以上节点，**consensus** 值为 token + 2000 毫秒。

如果 **cman** 工具以这种方式配置您的 consensus 超时，那么将来您从两个节点移动到三个（或者更多）节点就需要重启集群，因为 consensus 超时需要根据 token 超时改为较大的值。

如果您要配置一个双节点集群，并想要在将来升级到两个以上节点，您可以覆盖这个 consensus 超时以便当从两个节点移动到到三个（或者更多）节点时不需要重启集群。您可以在 **cluster.conf** 中进行如下操作以达到此目的：

```
<totem token="X" consensus="X + 2000" />
```

注：这个配置解析程序不会自动计算 X+ 2000。您在此必须使用整数而不是等式。

在双节点集群中使用优化的 consensus 超时的优点是总体上降低了双节点故障切换耗时，因为 consensus 不是 token 超时的功能。

注：在 **cman** 的双节点自动探测中，物理节点数是那些有用的节点而不是 **cluster.conf** 文件中出现的 **two_node=1** 指令。

7.3. 配置 FENCING

配置 fencing 包括（a）在一个集群中配置一个或者多个 fence 设备；（b）为每个节点配置一个或者多个 fence 方法（使用指定的一个或者多个 fence 设备）。

根据您的配置所需 fence 设备类型和 fence 方法配置 **cluster.conf**，如下：

1. 在 **fencedevices** 部分，使用 **fencedevice** 元素和 fence 设备独立属性指定每个 fence 设备。例 7.3 “添加到 **cluster.conf** 中的 APC Fence 设备” 演示添加了 APC fence 设备的配置文件示例。
2. 在 **clusternodes** 部分，每个 **clusternodes** 部分的 **fence** 元素中，指定每个节点的 fence 方法。使用 **method** 属性 **name** 指定 fence 方法名称。使用 **device** 元素及其属性 **name** 和具体 fence 设备参数为每个 fence 方法指定 fence 设备。例 7.4 “添加到 **cluster.conf** 的 fence 方法” 演示了集群中每个节点一个 fence 设备的 fence 方法示例。
3. 在非电源 fence 方法（即 SAN/存储 fencing）的 **clusternodes** 部分添加 **unfence** 字段。这可保证在重启该节点前不会重新启用被 fence 的节点。有关 unfencing 节点的详情请参考 **fence_node**(8) man page。

   与 **fence** 部分不同，**unfence** 部分不包含 **method**。它直接包含 **device** 参考，使用 "on" 或者 "enable" 的明确动作（**action**）成为 **fence** 对应设备部分的镜像。**fence** 和 **unfence device** 行都参考同一 **fencedevice**，并应在每个节点中重复同样的参数。

将 **action** 属性指定为 "on" 或者 "enable"，可在重启时启用该节点。例 7.4 “添加到 **cluster.conf** 的 fence 方法” 和 例 7.5 “**cluster.conf**：每个节点中有多种 Fence 方法” 中包含 **unfence** 元素和属性示例。

有关 **unfence** 详情请参考 **fence_node** man page。

4. 增加 **config_version** 属性参数即可更新该参数（例如：从 **config_version="2"** 改为 **config_version="3">**）。
5. 保存 **/etc/cluster/cluster.conf**。
6. **（可选）**，运行 **ccs_config_validate** 命令，确认根据集群方案（**cluster.rng**）更新的文件。例如：

```
[root@example-01 ~]# ccs_config_validate
Configuration validates
```

7. 请运行 **cman_tool version -r** 命令在剩余的所有集群节点中推广。这还将运行附加验证。需要在每个集群节点中都运行 **ricci** 方可推广更新的集群配置信息。
8. 确认推广了更新的文件。
9. 执行 第 7.4 节 “配置故障切换域”。

如果需要，您可以通过在每个节点中使用多种 fence 方法和在每个 fence 方法中使用多个 fence 设备配置复杂配置。当在每个节点中指定多个 fence 方法时，如果使用第一个方法 **fenced** 执行 fence 操作失败，则 fence 守护进程会尝试下一个方法，然后继续循环尝试所有方法直到成功为止。

有时 fencing 节点需要禁用两个 I/O 路径或者两个电源端口。这可通过在 fence 方法中指定两个或者多个设备完成。**fenced** 为每个 fence 设备行运行一次 fence 代理；只有全部成功方可认为是成功。

“Fencing 配置示例”一节 中演示了更复杂的配置。

您可以在 fence 设备代理 man page 中找到有关配置具体 fence 设备的更多信息（例如：**fence_apc** man page）。另外，您可以在 附录 A, *Fence 设备参数* 中获得有关 fence 参数的更多信息，在 **/usr/sbin/** 中获得 fence 代理的更多信息，在 **/usr/share/cluster/cluster.rng** 中获得有关集群方案的更多信息，在 **/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html** 中获得有关注释方案的更多信息（例如：**/usr/share/doc/cman-3.0.12/cluster_conf.html**）。

Fencing 配置示例

以下示例演示了每个节点一个 fence 方法以及每个 fence 方法一个 fence 设备的简单配置：

- 例 7.3 “添加到 **cluster.conf** 中的 APC Fence 设备”
- 例 7.4 “添加到 **cluster.conf** 的 fence 方法”

以下示例演示了较复杂的配置：

- 例 7.5 “**cluster.conf**：每个节点中有多种 Fence 方法”
- 例 7.6 “**cluster.conf**：Fencing，多路径多端口”
- 例 7.7 “**cluster.conf**：使用双电源 Fencing 节点”

注意

本节中的示例并不完全，即还有其他方法根据您的要求配置 fencing。

例 7.3. 添加到 cluster.conf 中的 APC Fence 设备

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc" passwd="password_example"/>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

在这个示例中，已将 fence 设备（**fencedevice**）添加到 **fencedevices** 元素中，指定 fence 代理（**agent**）为 **fence_apc**，IP 地址（**ipaddr**）为 **apc_ip_example**，登录（**login**）为 **login_example**，fence 设备名称（**name**）为 **apc**，以及密码（**passwd**）为 **password_example**。

例 7.4. 添加到 cluster.conf 的 fence 方法

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
```

```
          </fence>
      </clusternode>
      <clusternode name="node-03.example.com" nodeid="3">
          <fence>
             <method name="APC">
               <device name="apc" port="3"/>
             </method>
          </fence>
      </clusternode>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc" passwd="password_example"/>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

在这个示例中为每个节点添加了 fence 方法（**method**）。每个节点的 fence 方法名称（**name**）为 **APC**。将用于每个节点中 fence 方法的设备（**device**）名称（**name**）指定为 **apc**，并为每个节点指定唯一 APC 切换电源端口号（**port**）。例如：node-01.example.com 的端口号为 **1**（**port="1"**）。每个节点的设备名称（**device name="apc"**）根据 **fencedevices** 元素 **apc** 行中的名称（**name**）指向 fence 设备：**fencedevice agent="fence_apc" ipaddr="apc_ip_example" login="login_example" name="apc" passwd="password_example"**。

例 7.5. cluster.conf：每个节点中有多种 Fence 方法

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
            <method name="SAN">
       <device name="sanswitch1" port="11"/>
             </method>
         </fence>
         <unfence>
             <device name="sanswitch1" port="11" action="on"/>
         </unfence
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
            <method name="SAN">
       <device name="sanswitch1" port="12"/>
             </method>
         </fence>
```

```
          <unfence>
              <device name="sanswitch1" port="12" action="on"/>
          </unfence
      </clusternode>
      <clusternode name="node-03.example.com" nodeid="3">
          <fence>
             <method name="APC">
               <device name="apc" port="3"/>
             </method>
             <method name="SAN">
        <device name="sanswitch1" port="13"/>
              </method>
          </fence>
          <unfence>
              <device name="sanswitch1" port="13" action="on"/>
          </unfence
      </clusternode>
   </clusternodes>
   <fencedevices>
        <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc" passwd="password_example"/>
        <fencedevice agent="fence_sanbox2" ipaddr="san_ip_example"
login="login_example" name="sanswitch1" passwd="password_example"/>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

例 7.6. `cluster.conf`：Fencing，多路径多端口

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
          <fence>
             <method name="SAN-multi">
        <device name="sanswitch1" port="11"/>
        <device name="sanswitch2" port="11"/>
      </method>
          </fence>
          <unfence>
              <device name="sanswitch1" port="11" action="on"/>
              <device name="sanswitch2" port="11" action="on"/>
          </unfence
      </clusternode>
      <clusternode name="node-02.example.com" nodeid="2">
          <fence>
             <method name="SAN-multi">
        <device name="sanswitch1" port="12"/>
        <device name="sanswitch2" port="12"/>
              </method>
          </fence>
```

```
          <unfence>
              <device name="sanswitch1" port="12" action="on"/>
              <device name="sanswitch2" port="12" action="on"/>
          </unfence
      </clusternode>
      <clusternode name="node-03.example.com" nodeid="3">
          <fence>
             <method name="SAN-multi">
        <device name="sanswitch1" port="13"/>
        <device name="sanswitch2" port="13"/>
             </method>
          </fence>
          <unfence>
              <device name="sanswitch1" port="13" action="on"/>
              <device name="sanswitch2" port="13" action="on"/>
          </unfence
      </clusternode>
   </clusternodes>
   <fencedevices>
        <fencedevice agent="fence_sanbox2" ipaddr="san_ip_example"
login="login_example" name="sanswitch1" passwd="password_example"/>
        <fencedevice agent="fence_sanbox2" ipaddr="san_ip_example"
login="login_example" name="sanswitch2" passwd="password_example"/>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

例 7.7. `cluster.conf`：使用双电源 Fencing 节点

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC-dual">
              <device name="apc1" port="1"action="off"/>
              <device name="apc2" port="1"action="off"/>
              <device name="apc1" port="1"action="on"/>
              <device name="apc2" port="1"action="on"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC-dual">
              <device name="apc1" port="2"action="off"/>
              <device name="apc2" port="2"action="off"/>
              <device name="apc1" port="2"action="on"/>
              <device name="apc2" port="2"action="on"/>
            </method>
         </fence>
```

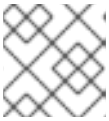
```
      </clusternode>
      <clusternode name="node-03.example.com" nodeid="3">
          <fence>
             <method name="APC-dual">
               <device name="apc1" port="3"action="off"/>
               <device name="apc2" port="3"action="off"/>
               <device name="apc1" port="3"action="on"/>
               <device name="apc2" port="3"action="on"/>
             </method>
          </fence>
      </clusternode>
   </clusternodes>
   <fencedevices>
       <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc1" passwd="password_example"/>
       <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc2" passwd="password_example"/>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

当使用电源切换 fence 使用双电源的节点时，必须告知代理在恢复两个电源端口前关闭两个电源端口。默认代理开关行为将导致永远无法在该节点完全禁用该电源。

7.4. 配置故障切换域

故障切换域是一个命名的集群节点子集，它可在节点失败事件中运行集群服务。故障切换域有以下特征：

- 无限制 — 允许您为在子集指定首选成员，但分配给这个域名的集群服务可在任意可用成员中运行。
- 限制 — 允许您限制可运行具体集群服务的成员。如果在限制故障切换域中没有可用成员，则无法启动集群服务（手动或者使用集群软件均不可行）。
- 无序 — 当将集群服务分配给一个无序故障切换域时，则可从可用故障切换域成员中随机选择运行集群服务的成员，没有优先顺序。
- 有序 — 允许您在故障切换域成员中指定首选顺序。有序故障切换域从优先数字最小的节点开始选择。即故障切换域中优先数字为 "1" 的节点具有最高的优先权，因此它是故障切换域中的首选节点。那个节点之后，第二首选节点应为含有下一个最高优先权数字的节点，依此类推。
- 故障恢复 — 允许您指定在故障切换域中的服务是否应该恢复到节点失败前最初运行的节点。配置这个特性在作为有序故障切换域一部分节点重复失败的环境中很有帮助。在那种情况下，如果某个节点是故障切换域中的首选节点，在可能在首选节点和其它节点间重复切换和恢复某个服务，从而不会对性能产生严重影响。



注意

故障恢复特性只适用于配置了有序故障切换的集群。

注意

更改故障切换域配置对目前运行中的服务无效。

注意

操作*不需要*的故障切换域。

默认情况下故障切换域为无限制和无序的。

在由几个成员组成的集群中，使用限制故障切换域可最大程度降低设置集群以便运行集群服务的工作（比如 **httpd**），它要求您在运行该集群服务的所有成员中进行完全一致的配置。您不需要将整个集群设置为运行该集群服务，只要设置与该集群服务关联的限制故障切换域中的成员即可。

注意

要配置首选成员，您可以创建一个只由一个集群成员组成的无限故障切换域。这样可让该集群服务在那个主要集群成员（首选成员）中运行，但允许将集群服务切换到任意其它成员中。

使用以下步骤配置故障切换域：

1. 在集群的任意节点中打开 **/etc/cluster/cluster.conf**。
2. 为每个要使用的故障切换域 **rm** 元素中添加以下框架部分：

```
        <failoverdomains>
            <failoverdomain name="" nofailback="" ordered=""
restricted="">
                <failoverdomainnode name="" priority=""/>
                <failoverdomainnode name="" priority=""/>
                <failoverdomainnode name="" priority=""/>
            </failoverdomain>
        </failoverdomains>
```

注意

failoverdomainnode 属性的数目由故障切换域中的节点数决定。前面文本中的框架 **failoverdomainnode** 部分有三个 **failoverdomainnode** 元素（没有指定节点名称），表示在故障切换域中有三个节点。

3. 在 **failoverdomainnode** 部分提供了元素和属性值。有关元素和属性描述请参考只是集群方案的 *failoverdomain* 部分。注释集群方案位于任意集群节点中的 **/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html**（例如：**/usr/share/doc/cman-3.0.12/cluster_conf.html**）。有关 **failoverdomains** 部分示例请参考 例 7.8 “在 **cluster.conf** 中添加故障切换域”。
4. 增加 **config_version** 属性参数即可更新该参数（例如：从 **config_version="2"** 改为 **config_version="3">**）。

5. 保存 **/etc/cluster/cluster.conf**。

6. （可选），运行 **ccs_config_validate** 命令，根据集群方案（**cluster.rng**）验证该文件。例如：

```
[root@example-01 ~]# ccs_config_validate
Configuration validates
```

7. 运行 **cman_tool version -r** 命令在其他集群节点中推广该配置。

8. 执行 第 7.5 节 “配置 HA 服务”。

例 7.8 “在 **cluster.conf** 中添加故障切换域” 演示使用有序、无限故障切换域配置示例。

例 7.8. 在 cluster.conf 中添加故障切换域

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
            <method name="APC">
              <device name="apc" port="3"/>
            </method>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc" passwd="password_example"/>
   </fencedevices>
   <rm>
       <failoverdomains>
           <failoverdomain name="example_pri" nofailback="0"
ordered="1" restricted="0">
               <failoverdomainnode name="node-01.example.com"
priority="1"/>
               <failoverdomainnode name="node-02.example.com"
priority="2"/>
               <failoverdomainnode name="node-03.example.com"
priority="3"/>
```

```
            </failoverdomain>
        </failoverdomains>
    </rm>
</cluster>
```

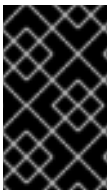
failoverdomains 部分包含集群中每个故障切换域中的 **failoverdomains** 部分。这个示例中有一个故障切换域。在 **failoverdomains** 行中，名称（**name**）为 **example_pri**。另外，它指定无故障恢复（**failback="0"**），即故障切换为有序的（**ordered="1"**），故障切换域为无限的（**restricted="0"**）。

7.5. 配置 HA 服务

配置 HA（高可用性）服务包括配置资源以及为服务分配资源。

以下小节描述了如何编辑 **/etc/cluster/cluster.conf** 添加资源和服务。

- 第 7.5.1 节 “添加集群资源”
- 第 7.5.2 节 “在集群中添加集群服务”



重要

配置高可用性资源和服务有很多可能性。要更好了解资源参数和资源行为，请参考 附录 B, *HA 资源参数* 和 附录 C, *HA 资源行为*。为优化性能并保证您的配置可被支持，请联络授权红帽支持代表。

7.5.1. 添加集群资源

您可配置两种类型的资源：

- 全局 — 集群中的任何服务都可用的资源。这些在配置文件的 **resources** 部分配置（**rm** 元素中）。
- 具体服务 — 只在一个服务中可用的资源。这些在配置文件的每个 **service** 部分配置（在 **rm** 元素中）。

本小节描述了如何添加全局资源。有关配置具体服务资源的步骤请参考 第 7.5.2 节 “在集群中添加集群服务”。

要添加全局集群资源，请按照本小节中的步骤执行。

1. 在集群的任意节点中打开 **/etc/cluster/cluster.conf**。
2. 在 **rm** 元素中添加 **resources** 部分。例如：

```
<rm>
    <resources>

    </resources>
</rm>
```

3. 根据您要创建的服务为其部署资源。例如：这里是 Apache 服务中要使用的资源。它们包括一个文件系统（**fs**）资源、一个 IP（**ip**）资源和一个 Apache（**apache**）资源。

```
    <rm>
        <resources>
           <fs name="web_fs" device="/dev/sdd2"
mountpoint="/var/www" fstype="ext3"/>
           <ip address="127.143.131.100" monitor_link="yes"
sleeptime="10"/>
           <apache config_file="conf/httpd.conf"
name="example_server" server_root="/etc/httpd" shutdown_wait="0"/>
        </resources>
    </rm>
```

   例 7.9 “**cluster.conf** 添加了资源的文件” 演示了添加 **resources** 部分的 **cluster.conf** 文件示例。

4. 通过增加其值更新 **config_version** 属性（例如：从 **config_version="2"** 改为 **config_version="3"**）。

5. 保存 **/etc/cluster/cluster.conf**。

6. **（可选）**，运行 **ccs_config_validate** 命令，根据集群方案（**cluster.rng**）验证该文件。例如：

```
[root@example-01 ~]# ccs_config_validate
Configuration validates
```

7. 运行 **cman_tool version -r** 命令在其他集群节点中推广该配置。

8. 确认推广了更新的文件。

9. 执行 第 7.5.2 节 “在集群中添加集群服务”。

例 7.9. cluster.conf 添加了资源的文件

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
         </fence>
     </clusternode>
```

```
      <clusternode name="node-03.example.com" nodeid="3">
          <fence>
             <method name="APC">
               <device name="apc" port="3"/>
             </method>
          </fence>
      </clusternode>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc" passwd="password_example"/>
   </fencedevices>
   <rm>
       <failoverdomains>
           <failoverdomain name="example_pri" nofailback="0"
ordered="1" restricted="0">
               <failoverdomainnode name="node-01.example.com"
priority="1"/>
               <failoverdomainnode name="node-02.example.com"
priority="2"/>
               <failoverdomainnode name="node-03.example.com"
priority="3"/>
           </failoverdomain>
       </failoverdomains>
       <resources>
           <fs name="web_fs" device="/dev/sdd2" mountpoint="/var/www"
fstype="ext3"/>
           <ip address="127.143.131.100" monitor_link="yes"
sleeptime="10"/>
           <apache config_file="conf/httpd.conf" name="example_server"
server_root="/etc/httpd" shutdown_wait="0"/>
        </resources>

   </rm>
</cluster>
```

7.5.2. 在集群中添加集群服务

要在集群中添加集群服务，请按照本小节中的步骤执行。

1. 在集群的任意节点中打开 **/etc/cluster/cluster.conf**。
2. 为每个服务在 **rm** 元素中添加 **service** 部分。例如：

```
    <rm>
        <service autostart="1" domain="" exclusive="0" name=""
recovery="restart">

        </service>
    </rm>
```

3. 在 **service** 元素中配置以下参数（属性）：

   - **autostart** — 指定是否在集群启动时自动启动该服务。使用‘1’启用，‘0’禁用，默认为启用。
   - **domain** — 指定故障切换域（如果需要）。
   - **exclusive** — 指定该服务只在没有其它服务运行的节点中的策略。
   - **recovery** — 为该服务指定恢复策略。选项为 relocate、restart、disable 或者 restart-disable 该服务。

4. 根据您要使用的资源类型使用全局或者具体服务资源部署该服务。

   例如：这里是使用全局资源的 Apache 服务：

```
    <rm>
        <resources>
                <fs name="web_fs" device="/dev/sdd2"
mountpoint="/var/www" fstype="ext3"/>
                <ip address="127.143.131.100" monitor_link="yes"
sleeptime="10"/>
                <apache config_file="conf/httpd.conf"
name="example_server" server_root="/etc/httpd" shutdown_wait="0"/>
        </resources>
        <service autostart="1" domain="example_pri" exclusive="0"
name="example_apache" recovery="relocate">
                <fs ref="web_fs"/>
                <ip ref="127.143.131.100"/>
                <apache ref="example_server"/>
        </service>
    </rm>
```

   例如：这里是具体服务资源使用的 Apache 服务：

```
    <rm>
        <service autostart="0" domain="example_pri" exclusive="0"
name="example_apache2" recovery="relocate">
                <fs name="web_fs2" device="/dev/sdd3"
mountpoint="/var/www2" fstype="ext3"/>
                <ip address="127.143.131.101" monitor_link="yes"
sleeptime="10"/>
                <apache config_file="conf/httpd.conf"
name="example_server2" server_root="/etc/httpd" shutdown_wait="0"/>
        </service>
    </rm>
```

   例 7.10 “添加了服务的 **cluster.conf**：一个使用全局资源，一个使用具体服务资源” 演示有两个服务的 **cluster.conf** 文件示例：

- **example_apache** — 这个服务使用全局资源 **web_fs**、**127.143.131.100** 和 **example_server**。
- **example_apache2** — 这个服务使用具体服务资源 **web_fs2**、**127.143.131.101** 和 **example_server2**。

5. 增加 **config_version** 属性参数即可更新该参数（例如：从 **config_version="2"** 改为 **config_version="3">**）。
6. 保存 **/etc/cluster/cluster.conf**。
7. **（可选）**，运行 **ccs_config_validate** 命令，确认根据集群方案（**cluster.rng**）更新的文件。例如：

```
[root@example-01 ~]# ccs_config_validate
Configuration validates
```

8. 运行 **cman_tool version -r** 命令在其他集群节点中推广该配置。
9. 确认推广了更新的文件。
10. 执行 第 7.8 节 “验证配置”。

例 7.10. 添加了服务的 cluster.conf：一个使用全局资源，一个使用具体服务资源

```
<cluster name="mycluster" config_version="3">
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
            <method name="APC">
              <device name="apc" port="3"/>
            </method>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc" passwd="password_example"/>
   </fencedevices>
   <rm>
```

```
        <failoverdomains>
            <failoverdomain name="example_pri" nofailback="0"
ordered="1" restricted="0">
                <failoverdomainnode name="node-01.example.com"
priority="1"/>
                <failoverdomainnode name="node-02.example.com"
priority="2"/>
                <failoverdomainnode name="node-03.example.com"
priority="3"/>
            </failoverdomain>
        </failoverdomains>
        <resources>
            <fs name="web_fs" device="/dev/sdd2" mountpoint="/var/www"
fstype="ext3"/>
            <ip address="127.143.131.100" monitor_link="yes"
sleeptime="10"/>
            <apache config_file="conf/httpd.conf" name="example_server"
server_root="/etc/httpd" shutdown_wait="0"/>
        </resources>
        <service autostart="1" domain="example_pri" exclusive="0"
name="example_apache" recovery="relocate">
            <fs ref="web_fs"/>
            <ip ref="127.143.131.100"/>
            <apache ref="example_server"/>
        </service>
        <service autostart="0" domain="example_pri" exclusive="0"
name="example_apache2" recovery="relocate">
            <fs name="web_fs2" device="/dev/sdd3" mountpoint="/var/www2"
fstype="ext3"/>
            <ip address="127.143.131.101" monitor_link="yes"
sleeptime="10"/>
            <apache config_file="conf/httpd.conf" name="example_server2"
server_root="/etc/httpd" shutdown_wait="0"/>
        </service>
    </rm>
</cluster>
```

7.6. 配置冗余环协议

从红帽企业版 Linux 6.4 开始，红帽高可用附加组件支持冗余环协议配置。

当将系统配置为使用冗余环协议时，必须考虑以下问题：

- 不要指定两个以上的环。
- 每个环必须使用相同的协议；不要混合 IPv4 和 IPv6。
- 如果有必要，您可以手动为第二个环指定多播地址。如果您为第二个环指定多播地址，要么使用备用多播地址，要么备用端口必须与第一个环的多播地址不同。如果您没有指定备用的多播地址，则系统将自动为第二个环使用不同的多播地址。

  如果您指定备用端口，则第一个环的端口号必须与第一个环的端口号有两个以上的差别，因为系统本身使用端口即端口-1 执行操作。

- 不要在同一子网中使用不同的接口。
- 一般最好在两个不同的 NIC 和两个不同的切换中配置冗余环协议，以防一个 NIC 或者切换失败。
- 不要使用 **ifdown** 命令或者 **service network stop** 模仿网络失败。这会破坏整个集群并需要您重启集群中的所有节点方可恢复。
- 不要使用 **NetworkManager**，因为它会在拔掉电缆时执行 **ifdown** 命令。
- 当一个 NIC 失败时，会将整个环标记为失败。
- 恢复失败的环不需要人工介入。要恢复冗余环则需要修复造成失败的最初原因，比如失败的 NIC 或者切换。

要为冗余环协议指定辅网络接口，您可以在 **cluster.conf** 配置文件的 **clusternode** 部分添加 **altname** 组件。指定 **altname** 后，您必须指定 **name** 属性为该节点指出辅主机名或者 IP 地址。

以下示例指定 **clusternet-node1-eth2** 作为集群节点 **clusternet-node1-eth1** 的备用名称。

```
<cluster name="mycluster" config_version="3" >
  <logging debug="on"/>
  <clusternodes>
    <clusternode name="clusternet-node1-eth1" votes="1" nodeid="1">
      <fence>
        <method name="single">
          <device name="xvm" domain="clusternet-node1"/>
        </method>
      </fence>
      <altname name="clusternet-node1-eth2"/>
    </clusternode>
```

clusternode 中的 **altname** 部分并不需要固定的位置。它可以在 **fence** 部分前面，也可以在它后面。不要为一个集群节点指定一个以上 **altname**，否则系统将无法启动。

另外，您可以为第二个环手动指定多播地址、端口以及 TTL，方法是在 **cluster.conf** 配置文件的 **cman** 部分添加一个 **altmulticast** 组件。**altmulticast** 组件可以是 **addr**、**port** 和 **ttl** 参数。

以下示例显示集群配置文件的 **cman** 部分，该部分为第二个环设置多播地址、端口和 TTL。

```
<cman>
   <multicast addr="239.192.99.73" port="666" ttl="2"/>
   <altmulticast addr="239.192.99.88" port="888" ttl="3"/>
</cman>
```

7.7. 配置 DEBUG 选项

您可以为集群中的所有守护进程启用 debugging，或者您可以为具体集群进程启用日志。

要为所有守护进程启用 debugging，请在 **/etc/cluster/cluster.conf** 中添加以下行。默认情况下会将日志指向 **/var/log/cluster/*daemon*.log** 文件。

```
<cluster config_version="7" name="rh6cluster">
  <logging debug="on"/>
   ...
</cluster>
```

要为独立集群进程启用 debugging，请在 **/etc/cluster/cluster.conf** 文件中添加以下行。每个守护进程的日志配置可覆盖全局设置。

```
<cluster config_version="7" name="rh6cluster">
   ...
  <logging>
       <!-- turning on per-subsystem debug logging -->
       <logging_daemon name="corosync" debug="on" />
       <logging_daemon name="fenced" debug="on" />
       <logging_daemon name="qdiskd" debug="on" />
       <logging_daemon name="rgmanager" debug="on" />
       <logging_daemon name="dlm_controld" debug="on" />
       <logging_daemon name="gfs_controld" debug="on" />
  </logging>
   ...
</cluster>
```

对于您可以启用日志的日志守护进程列表以及您可以同时配置全局和按守护进程记录日志的附加日志选项，请参考 **cluster.conf**(5) man page。

7.8. 验证配置

您创建集群配置文件后，请执行以下步骤确认其正常运行：

1. 在每个节点中重启集群软件。这个动作确保在运行的配置中包括只在启动时检查的配置添加。您可以运行 **service cman restart** 命令重启集群软件。例如：

```
[root@example-01 ~]# service cman restart
Stopping cluster:
   Leaving fence domain...                                 [  OK
]
   Stopping gfs_controld...                                [  OK
]
   Stopping dlm_controld...                                [  OK
]
   Stopping fenced...                                      [  OK
]
   Stopping cman...                                        [  OK
]
   Waiting for corosync to shutdown:                       [  OK  ]
   Unloading kernel modules...                             [  OK
]
   Unmounting configfs...                                  [  OK
]
Starting cluster:
```

```
   Checking Network Manager...                                [  OK
]
   Global setup...                                            [  OK
]
   Loading kernel modules...                                  [  OK
]
   Mounting configfs...                                       [  OK
]
   Starting cman...                                           [  OK
]
   Waiting for quorum...                                      [  OK
]
   Starting fenced...                                         [  OK
]
   Starting dlm_controld...                                   [  OK
]
   Starting gfs_controld...                                   [  OK
]
   Unfencing self...                                          [  OK
]
   Joining fence domain...                                    [  OK
]
```

2. 如果使用 CLVM 创建集群的卷，则运行 **service clvmd start**。例如：

```
[root@example-01 ~]# service clvmd start
Activating VGs:                                               [  OK
]
```

3. 如果您使用 Red Hat GFS2，请运行 **service gfs2 start**。例如：

```
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                         [  OK  ]
Mounting GFS2 filesystem (/mnt/gfsB):                         [  OK  ]
```

4. 如果您使用高可用性（HA）服务，请运行 **service rgmanager start**。例如：

```
[root@example-01 ~]# service rgmanager start
Starting Cluster Service Manager:                             [  OK  ]
```

5. 在任意集群节点中运行 **cman_tool nodes**，确认那些节点作为集群的成员运行（在状态列 "Sts" 中被标记为 "M"）。例如：

```
[root@example-01 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M    548   2010-09-28 10:52:21  node-01.example.com
   2   M    548   2010-09-28 10:52:21  node-02.example.com
   3   M    544   2010-09-28 10:52:21  node-03.example.com
```

6. 在任意节点中使用 **clustat** 程序确认那些 HA 服务正常运行。另外 **clustat** 可显示集群节点状态。例如：

```
[root@example-01 ~]#clustat
```

```
Cluster Status for mycluster @ Wed Nov 17 05:40:00 2010
Member Status: Quorate

 Member Name                                ID   Status
 ------ ----                                ---- ------
 node-03.example.com                            3 Online, rgmanager
 node-02.example.com                            2 Online, rgmanager
 node-01.example.com                            1 Online, Local,
rgmanager

 Service Name                      Owner (Last)
State
 ------- ----                      ----- ------                    --
---
 service:example_apache            node-01.example.com
started
 service:example_apache2           (none)
disabled
```

7. 如果集群正常运行，则您完成了配置文件创建。您可使用命令行工具管理集群，如 第 8 章 *使用命令行工具管理红帽高可用性附加组件* 所述。

第 8 章 使用命令行工具管理红帽高可用性附加组件

本章论述了管理红帽高可用性附加组件的各种管理任务，它由以下小节组成：

- 第 8.1 节 “启动和停止集群软件”
- 第 8.2 节 “删除或者添加节点”
- 第 8.3 节 “管理高可用性服务”
- 第 8.4 节 “更新配置”

重要

确定您部署的红帽高可用性附加组件满足您的需要并可被支持。部署前请咨询授权红帽代表确认您的配置。另外请预留充分时间测试失败模式。

重要

本章通常使用 **cluster.conf** 元素和属性作为参考。有关 **cluster.conf** 元素和属性的完整列表，请参考 **/usr/share/cluster/cluster.rng** 中的集群方案，注释的方案位于 **/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html**（例如：**/usr/share/doc/cman-3.0.12/cluster_conf.html**）。

重要

本章的某些步骤调用 **cman_tool version -r** 命令在集群中传推广群配置。使用该命令需要运行 **ricci**。

注意

本章中的步骤可能包括某些命令行工具的具体命令，如 附录 E, *命令行工具小结* 所示。有关所有命令和变量的详情请参考每个命令行工具的 man page。

8.1. 启动和停止集群软件

您可以如 第 8.1.1 节 “启动集群软件” 和 第 8.1.2 节 “停止集群软件” 所述在某个节点中启动或者停止集群软件。在节点中启动集群软件可让该软件加入集群，在节点中停止集群软件则会让该软件离开集群。

8.1.1. 启动集群软件

要在节点中启动集群软件，请按以下顺序输入命令：

1. **service cman start**
2. 如果使用 CLVM 创建集群的卷，则请使用**service clvmd start**。
3. 如果您使用 Red Hat GFS2，则请使用 **service gfs2 start**。
4. 如果您使用高可用性（HA）服务（**rgmanager**），则请使用 **service rgmanager start**。

例如：

-

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                             [  OK  ]
   Global setup...                                         [  OK  ]
   Loading kernel modules...                               [  OK  ]
   Mounting configfs...                                    [  OK  ]
   Starting cman...                                        [  OK  ]
   Waiting for quorum...                                   [  OK  ]
   Starting fenced...                                      [  OK  ]
   Starting dlm_controld...                                [  OK  ]
   Starting gfs_controld...                                [  OK  ]
   Unfencing self...                                       [  OK  ]
   Joining fence domain...                                 [  OK  ]
[root@example-01 ~]# service clvmd start
Starting clvmd:                                            [  OK  ]
Activating VG(s):   2 logical volume(s) in volume group "vg_example" now
active
                                                           [  OK  ]
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                      [  OK  ]
Mounting GFS2 filesystem (/mnt/gfsB):                      [  OK  ]
[root@example-01 ~]# service rgmanager start
Starting Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]#
```

8.1.2. 停止集群软件

要在节点中停止集群软件，请按以下顺序输入命令：

1. 如果您使用高可用性（HA）服务（**rgmanager**），在请使用 **service rgmanager stop**。
2. 如果您使用 Red Hat GFS2，则请使用 **service gfs2 stop**。
3. 如果您将 **rgmanager** 与 Red Hat GFS2 一同使用，则请使用 **umount -at gfs2**，以便保证同时卸载了在启动 **rgmanager** 过程中（但不在关机过程中卸载）挂载的所有 GFS2 文件。
4. 如果使用 CLVM 创建集群的卷，则请使用 **service clvmd stop**。
5. **service cman stop**

例如：

```
[root@example-01 ~]# service rgmanager stop
Stopping Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]# service gfs2 stop
Unmounting GFS2 filesystem (/mnt/gfsA):                    [  OK  ]
Unmounting GFS2 filesystem (/mnt/gfsB):                    [  OK  ]
[root@example-01 ~]# umount -at gfs2
[root@example-01 ~]# service clvmd stop
Signaling clvmd to exit                                    [  OK  ]
clvmd terminated                                           [  OK  ]
[root@example-01 ~]# service cman stop
Stopping cluster:
   Leaving fence domain...                                 [  OK  ]
   Stopping gfs_controld...                                [  OK  ]
```

```
   Stopping dlm_controld...                                [  OK  ]
   Stopping fenced...                                      [  OK  ]
   Stopping cman...                                        [  OK  ]
   Waiting for corosync to shutdown:                       [  OK  ]
   Unloading kernel modules...                             [  OK  ]
   Unmounting configfs...                                  [  OK  ]
[root@example-01 ~]#
```

注意

在节点中停止集群软件可将其 HA 服务切换到另一个节点中。备选的方法是在停止集群软件前将 HA 服务重新定位或者迁移到另一个节点中。有关管理 HA 服务的详情请参考 第 8.3 节 “管理高可用性服务”。

8.2. 删除或者添加节点

本小节论述了如何从集群中删除节点或者在集群中添加节点。您可以从集群中删除节点，如 第 8.2.1 节 “从集群中删除节点” 所述，或者在集群中添加节点，如 第 8.2.2 节 “在集群中添加节点” 所述。

8.2.1. 从集群中删除节点

从集群中删除节点包括在节点中关闭要删除的集群软件，并更新集群配置以反映此变化。

重要

如果从集群中删除节点让该集群中只剩下两个节点，则您必须在更新集群配置文件后在每个节点中重启该集群软件。

要从集群中删除节点，请按照以下步骤操作：

1. 在任意节点中使用 **clusvcadm** 程序重新定位、迁移或者停止要从该集群删除的节点中的运行的所有 HA 服务。有关使用 **clusvcadm** 的详情请参考 第 8.3 节 “管理高可用性服务”。

2. 在您要从集群中删除的节点中停止集群软件，如 第 8.1.2 节 “停止集群软件” 所述。例如：

```
[root@example-01 ~]# service rgmanager stop
Stopping Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]# service gfs2 stop
Unmounting GFS2 filesystem (/mnt/gfsA):                    [  OK  ]
Unmounting GFS2 filesystem (/mnt/gfsB):                    [  OK  ]
[root@example-01 ~]# service clvmd stop
Signaling clvmd to exit                                    [  OK
]
clvmd terminated                                           [  OK
]
[root@example-01 ~]# service cman stop
Stopping cluster:
   Leaving fence domain...                                 [  OK
]
   Stopping gfs_controld...                                [  OK
]
   Stopping dlm_controld...                                [  OK
]
```

```
   Stopping fenced...                                         [  OK
]
   Stopping cman...                                           [  OK
]
   Waiting for corosync to shutdown:                          [  OK  ]
   Unloading kernel modules...                                [  OK
]
   Unmounting configfs...                                     [  OK
]
[root@example-01 ~]#
```

3. 在集群的任意节点中编辑 **/etc/cluster/cluster.conf**，删除要删除节点的 **clusternode** 部分。例如：在 例 8.1 “三节点配置” 中，如果要删除 node-03.example.com，则要删除该节点的 **clusternode** 部分。如果删除节点让该集群只剩下两个节点，则您可在配置文件中添加以下行以便单一节点可维护仲裁（例如：如果一个节点失败）：

   <cman two_node="1" expected_votes="1"/>

   有关三节点配置和双节点配置对比请参考 第 8.2.3 节 “三节点和双节点配置示例”。

4. 增加 **config_version** 属性参数即可更新该参数（例如：从 **config_version="2"** 改为 **config_version="3">**）。

5. 保存 **/etc/cluster/cluster.conf**。

6. **（可选）**，运行 **ccs_config_validate** 命令，验证根据集群方案（**cluster.rng**）更新的文件。例如：

   ```
   [root@example-01 ~]# ccs_config_validate
   Configuration validates
   ```

7. 运行 **cman_tool version -r** 命令在其他集群节点中推广该配置。

8. 确认推广了更新的文件。

9. 如果集群节点计数由大于 2 变为等于 2，则您必须重启集群软件，如下：

   1. 在每个节点中如 第 8.1.2 节 “停止集群软件” 所述停止集群软件。例如：

      ```
      [root@example-01 ~]# service rgmanager stop
      Stopping Cluster Service Manager:                       [  OK
      ]
      [root@example-01 ~]# service gfs2 stop
      Unmounting GFS2 filesystem (/mnt/gfsA):                 [  OK
      ]
      Unmounting GFS2 filesystem (/mnt/gfsB):                 [  OK
      ]
      [root@example-01 ~]# service clvmd stop
      Signaling clvmd to exit                                 [
      OK  ]
      clvmd terminated                                        [
      OK  ]
      [root@example-01 ~]# service cman stop
      Stopping cluster:
         Leaving fence domain...                              [
      ```

```
OK  ]
   Stopping gfs_controld...                                [
OK  ]
   Stopping dlm_controld...                                [
OK  ]
   Stopping fenced...                                      [
OK  ]
   Stopping cman...                                        [
OK  ]
   Waiting for corosync to shutdown:                       [  OK
]
   Unloading kernel modules...                             [
OK  ]
   Unmounting configfs...                                  [
OK  ]
[root@example-01 ~]#
```

2. 在每个节点中如 第 8.1.1 节 “启动集群软件” 所述启动集群软件。例如：

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                             [
OK  ]
   Global setup...                                         [
OK  ]
   Loading kernel modules...                               [
OK  ]
   Mounting configfs...                                    [
OK  ]
   Starting cman...                                        [
OK  ]
   Waiting for quorum...                                   [
OK  ]
   Starting fenced...                                      [
OK  ]
   Starting dlm_controld...                                [
OK  ]
   Starting gfs_controld...                                [
OK  ]
   Unfencing self...                                       [
OK  ]
   Joining fence domain...                                 [
OK  ]
[root@example-01 ~]# service clvmd start
Starting clvmd:                                            [
OK  ]
Activating VG(s):   2 logical volume(s) in volume group
"vg_example" now active
                                                           [
OK  ]
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                      [  OK
]
Mounting GFS2 filesystem (/mnt/gfsB):                      [  OK
]
[root@example-01 ~]# service rgmanager start
```

```
Starting Cluster Service Manager:                          [  OK
]
[root@example-01 ~]#
```

3. 在任意集群节点中运行 **cman_tool nodes** 以确认那些节点是作为集群的成员运行（在状态列 "Sts" 中被标记为 "M"）。例如：

```
[root@example-01 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M    548   2010-09-28 10:52:21  node-01.example.com
   2   M    548   2010-09-28 10:52:21  node-02.example.com
```

4. 在任意节点中使用 **clustat** 程序确认那些 HA 服务正常运行。另外 **clustat** 可显示集群节点状态。例如：

```
[root@example-01 ~]#clustat
Cluster Status for mycluster @ Wed Nov 17 05:40:00 2010
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
 node-02.example.com                         2 Online, rgmanager
 node-01.example.com                         1 Online, Local,
rgmanager

 Service Name                   Owner (Last)
State
 ------- ----                   ----- ------
-----
 service:example_apache         node-01.example.com
started
 service:example_apache2        (none)
disabled
```

8.2.2. 在集群中添加节点

在集群中添加节点包括更新集群配置；在要添加的节点中使用更新的配置；以及在该节点中启动集群软件。要在集群中添加节点，请执行以下步骤：

1. 在集群的任意节点中编辑 **/etc/cluster/cluster.conf**，为要添加的节点添加 **clusternode** 部分。例如：在 例 8.2 “双节点配置” 中，如果要添加 node-03.example.com，则请为该节点添加 **clusternode** 部分。如果添加节点导致群集从双节点变为三个或者三个以上节点，请从 **/etc/cluster/cluster.conf** 中删除 **cman** 属性：

   - **cman two_node="1"**
   - **expected_votes="1"**

   有关三节点配置和双节点配置对比请参考 第 8.2.3 节 “三节点和双节点配置示例”。

2. 增加 **config_version** 属性参数即可更新该参数（例如：从 **config_version="2"** 改为 **config_version="3">**）。

3. 保存 **/etc/cluster/cluster.conf**。

4. **（可选）**，运行 **ccs_config_validate** 命令，验证根据集群方案（**cluster.rng**）更新的文件。例如：

```
[root@example-01 ~]# ccs_config_validate
Configuration validates
```

5. 运行 **cman_tool version -r** 命令在其他集群节点中推广该配置。

6. 确认推广了更新的文件。

7. 在每个要添加到集群的节点的 **/etc/cluster/** 中使用更新的配置文件。例如：**scp** 命令会将更新的配置文件发送到要添加到集群的每个节点中。

8. 如果节点计数由 2 变为大于 2，则您必须在现有集群节点中重启集群软件，如下：

   1. 在每个节点中如 第 8.1.2 节 “停止集群软件” 所述停止集群软件。例如：

```
[root@example-01 ~]# service rgmanager stop
Stopping Cluster Service Manager:                          [  OK
]
[root@example-01 ~]# service gfs2 stop
Unmounting GFS2 filesystem (/mnt/gfsA):                    [  OK
]
Unmounting GFS2 filesystem (/mnt/gfsB):                    [  OK
]
[root@example-01 ~]# service clvmd stop
Signaling clvmd to exit                                    [
OK  ]
clvmd terminated                                           [
OK  ]
[root@example-01 ~]# service cman stop
Stopping cluster:
   Leaving fence domain...                                 [
OK  ]
   Stopping gfs_controld...                                [
OK  ]
   Stopping dlm_controld...                                [
OK  ]
   Stopping fenced...                                      [
OK  ]
   Stopping cman...                                        [
OK  ]
   Waiting for corosync to shutdown:                       [  OK
]
   Unloading kernel modules...                             [
OK  ]
   Unmounting configfs...                                  [
OK  ]
[root@example-01 ~]#
```

   2. 在每个节点中如 第 8.1.1 节 “启动集群软件” 所述启动集群软件。例如：

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                             [
```

```
OK  ]
   Global setup...                                         [
OK  ]
   Loading kernel modules...                               [
OK  ]
   Mounting configfs...                                    [
OK  ]
   Starting cman...                                        [
OK  ]
   Waiting for quorum...                                   [
OK  ]
   Starting fenced...                                      [
OK  ]
   Starting dlm_controld...                                [
OK  ]
   Starting gfs_controld...                                [
OK  ]
   Unfencing self...                                       [
OK  ]
   Joining fence domain...                                 [
OK  ]
[root@example-01 ~]# service clvmd start
Starting clvmd:                                            [
OK  ]
Activating VG(s):   2 logical volume(s) in volume group
"vg_example" now active
                                                           [
OK  ]
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                      [  OK
]
Mounting GFS2 filesystem (/mnt/gfsB):                      [  OK
]
[root@example-01 ~]# service rgmanager start
Starting Cluster Service Manager:                          [  OK
]
[root@example-01 ~]#
```

9. 在每个要添加到集群的节点中如 第 8.1.1 节 “启动集群软件” 所述启动集群软件。例如：

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                         [  OK
]
   Global setup...                                     [  OK
]
   Loading kernel modules...                           [  OK
]
   Mounting configfs...                                [  OK
]
   Starting cman...                                    [  OK
]
   Waiting for quorum...                               [  OK
]
   Starting fenced...                                  [  OK
]
```

```
   Starting dlm_controld...                                [  OK
]
   Starting gfs_controld...                                [  OK
]
   Unfencing self...                                       [  OK
]
   Joining fence domain...                                 [  OK
]
[root@example-01 ~]# service clvmd start
Starting clvmd:                                            [  OK
]
Activating VG(s):   2 logical volume(s) in volume group "vg_example"
now active
                                                           [  OK
]
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                      [  OK  ]
Mounting GFS2 filesystem (/mnt/gfsB):                      [  OK  ]

[root@example-01 ~]# service rgmanager start
Starting Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]#
```

10. 在任意节点中使用 **clustat** 程序确认每个添加的节点正作为集群的一部分运行。例如：

```
[root@example-01 ~]#clustat
Cluster Status for mycluster @ Wed Nov 17 05:40:00 2010
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
 node-03.example.com                         3 Online, rgmanager
 node-02.example.com                         2 Online, rgmanager
 node-01.example.com                         1 Online, Local,
rgmanager

 Service Name                   Owner (Last)
State
 ------- ----                   ----- ------                    --
---
 service:example_apache         node-01.example.com
started
 service:example_apache2        (none)
disabled
```

有关使用 **clustat** 的详情请参考 第 8.3 节 “管理高可用性服务”。

另外，您可以使用 **cman_tool status** 确认节点投票、节点计数以及仲裁计数。例如：

```
[root@example-01 ~]#cman_tool status
Version: 6.2.0
Config Version: 19
Cluster Name: mycluster
Cluster Id: 3794
Cluster Member: Yes
```

```
Cluster Generation: 548
Membership state: Cluster-Member
Nodes: 3
Expected votes: 3
Total votes: 3
Node votes: 1
Quorum: 2
Active subsystems: 9
Flags:
Ports Bound: 0 11 177
Node name: node-01.example.com
Node ID: 3
Multicast addresses: 239.192.14.224
Node addresses: 10.15.90.58
```

11. 在任意节点中您可以使用 **clusvcadm** 程序将运行中的服务重新定位或者迁移到新添加的节点中。另外，您可以启用所有禁用的服务。有关使用 **clusvcadm** 详情请参考 第 8.3 节 “管理高可用性服务”。

8.2.3. 三节点和双节点配置示例

以下是三节点和双节点配置示例对比。

例 8.1. 三节点配置

```
<cluster name="mycluster" config_version="3">
   <cman/>
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
         </fence>
     </clusternode>
     <clusternode name="node-03.example.com" nodeid="3">
         <fence>
            <method name="APC">
              <device name="apc" port="3"/>
            </method>
         </fence>
     </clusternode>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc" passwd="password_example"/>
   </fencedevices>
```

```
   <rm>
       <failoverdomains>
           <failoverdomain name="example_pri" nofailback="0"
ordered="1" restricted="0">
               <failoverdomainnode name="node-01.example.com"
priority="1"/>
               <failoverdomainnode name="node-02.example.com"
priority="2"/>
               <failoverdomainnode name="node-03.example.com"
priority="3"/>
           </failoverdomain>
       </failoverdomains>
       <resources>
           <fs name="web_fs" device="/dev/sdd2" mountpoint="/var/www"
fstype="ext3"/>
           <ip address="127.143.131.100" monitor_link="yes"
sleeptime="10"/>
           <apache config_file="conf/httpd.conf" name="example_server"
server_root="/etc/httpd" shutdown_wait="0"/>
       </resources>
       <service autostart="0" domain="example_pri" exclusive="0"
name="example_apache" recovery="relocate">
           <fs ref="web_fs"/>
           <ip ref="127.143.131.100"/>
           <apache ref="example_server"/>
       </service>
       <service autostart="0" domain="example_pri" exclusive="0"
name="example_apache2" recovery="relocate">
           <fs name="web_fs2" device="/dev/sdd3" mountpoint="/var/www"
fstype="ext3"/>
           <ip address="127.143.131.101" monitor_link="yes"
sleeptime="10"/>
           <apache config_file="conf/httpd.conf" name="example_server2"
server_root="/etc/httpd" shutdown_wait="0"/>
       </service>
   </rm>
</cluster>
```

例 8.2. 双节点配置

```
<cluster name="mycluster" config_version="3">
   <cman two_node="1" expected_votes="1"/>
   <clusternodes>
     <clusternode name="node-01.example.com" nodeid="1">
         <fence>
            <method name="APC">
              <device name="apc" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node-02.example.com" nodeid="2">
         <fence>
```

```
            <method name="APC">
              <device name="apc" port="2"/>
            </method>
          </fence>
   </clusternodes>
   <fencedevices>
         <fencedevice agent="fence_apc" ipaddr="apc_ip_example"
login="login_example" name="apc" passwd="password_example"/>
   </fencedevices>
   <rm>
       <failoverdomains>
           <failoverdomain name="example_pri" nofailback="0"
ordered="1" restricted="0">
               <failoverdomainnode name="node-01.example.com"
priority="1"/>
               <failoverdomainnode name="node-02.example.com"
priority="2"/>
           </failoverdomain>
       </failoverdomains>
       <resources>
           <fs name="web_fs" device="/dev/sdd2" mountpoint="/var/www"
fstype="ext3"/>
           <ip address="127.143.131.100" monitor_link="yes"
sleeptime="10"/>
           <apache config_file="conf/httpd.conf" name="example_server"
server_root="/etc/httpd" shutdown_wait="0"/>
       </resources>
       <service autostart="0" domain="example_pri" exclusive="0"
name="example_apache" recovery="relocate">
           <fs ref="web_fs"/>
           <ip ref="127.143.131.100"/>
           <apache ref="example_server"/>
       </service>
       <service autostart="0" domain="example_pri" exclusive="0"
name="example_apache2" recovery="relocate">
           <fs name="web_fs2" device="/dev/sdd3" mountpoint="/var/www"
fstype="ext3"/>
           <ip address="127.143.131.101" monitor_link="yes"
sleeptime="10"/>
           <apache config_file="conf/httpd.conf" name="example_server2"
server_root="/etc/httpd" shutdown_wait="0"/>
       </service>
   </rm>
</cluster>
```

8.3. 管理高可用性服务

您可以使用 **Cluster Status Utility**，即 **clustat** 和 **Cluster User Service Administration Utility**，即 **clusvcadm** 管理高可用性服务。**clustat** 显示集群的状态，**clusvcadm** 提供管理高可用性服务的工具。

本小节提供有关使用 **clustat** 和 **clusvcadm** 命令管理 HA 服务的信息，它包括以下小节：

- 第 8.3.1 节 “使用 **clustat** 显示 HA 服务”
- 第 8.3.2 节 “使用 **clusvcadm** 管理 HA 服务”

8.3.1. 使用 clustat 显示 HA 服务

clustat 显示集群状态。它为您提供成员信息、仲裁查看、所有高可用性服务的状态，并给出运行 **clustat** 命令的节点（本地）。表 8.1 “服务状态” 描述运行 **clustat** 时会出现并显示的服务状态。例 8.3 “**clustat** 显示” 给出 **clustat** 显示示例。有关运行 **clustat** 命令的详情请参考 **clustat** man page。

表 8.1. 服务状态

服务状态	描述
「启动（Started）」	已配置该服务资源，且可用于拥有该服务的集群系统。
「恢复中（Recovering）」	该服务正在另一个节点中等待启动。
「禁用（Disabled）」	该服务已经被禁用，且没有分配拥有者。该集群永远不能自动重启禁用的服务。
「停止（Stopped）」	在停止状态中，将评价该服务以便在下一个服务或者节点过渡后启动。这是一个临时状态。您可以在这个状态中禁用或者启用该服务。
「失败（Failed）」	假设该服务已死。无论何时，当资源的 *stop* 操作失败时，服务都会处于这个状态。服务处于这个状态后，您在发出 **disable** 请求前必须确认没有为其分配任何资源（例如挂载的文件系统）。当某个服务处于这种状态时，唯一可行的操作就是 **disable**。
「未初始化（Uninitialized）」	在启动和运行 **clustat -f** 的过程中，有些时候可以出现这个状态。

例 8.3. clustat 显示

```
[root@example-01 ~]#clustat
Cluster Status for mycluster @ Wed Nov 17 05:40:15 2010
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
 node-03.example.com                         3 Online, rgmanager
 node-02.example.com                         2 Online, rgmanager
 node-01.example.com                         1 Online, Local,
rgmanager

 Service Name                   Owner (Last)                   State
 ------- ----                   ----- ------                   -----
 service:example_apache         node-01.example.com            started
 service:example_apache2        (none)
disabled
```

8.3.2. 使用 clusvcadm 管理 HA 服务

您可以使用 **clusvcadm** 命令管理 HA 服务。使用它您可以执行以下操作：

- 启用并启动服务。
- 禁用服务。
- 停止服务。
- 冻结服务
- 解冻服务
- 迁移服务（只用于虚拟机服务）
- 重新定位服务。
- 重启服务。

表 8.2 “服务操作” 详细描述了这些操作。有关如何执行那些操作的完整论述请参考 **clusvcadm** 程序 man page。

表 8.2. 服务操作

服务操作	描述	命令语法
「启用」	有条件地在首选对象中，根据故障切换域规则自选启动服务。二者缺一，则运行 **clusvcadm** 的本地主机将会启动该服务。如果原始*启动*失败，则该服务的行为会类似*重新定位*请求（请参考本表格中的**「重新定位」**）。如果操作成功，则该服务会处于启动的状态。	**clusvcadm -e <service_name>** 或者 **clusvcadm -e <service_name> -m <member>**（使用 -m 选项指定要启动该服务的首选目标成员。）
「禁用」	停止该服务使其处于禁用状态。当某个服务处于*失败*状态时，这是唯一允许的操作。	**clusvcadm -d <service_name>**
「重新定位」	将该服务移动到另一个节点中。您也可以指定首选节点接受此服务，但如果在那个主机中无法运行该服务（例如：如果服务无法启动或者主机离线），则无法阻止重新定位，并选择另一个节点。**rgmanager** 尝试在该集群的每个有权限的节点中启动该服务。如果集群中的没有任何有权限的目标可以成功启动该服务，则重新定位就会失败，同时会尝试在最初拥有者中重启该服务。如果原始拥有者无法重启该服务，则该服务会处于*停止*的状态。	**clusvcadm -r <service_name>** 或者 **clusvcadm -r <service_name> -m <member>**（使用 -m 选项指定要启动该服务的首选目标成员。）
「停止」	停止该服务并使其处于*停止*状态。	**clusvcadm -s <service_name>**

服务操作	描述	命令语法
「冻结」	在目前运行某个服务的节点中冻结该服务。这样会在节点失败事件中或者 rgmanager 停止时，阻止服务状态检查以及故障切换。这可用来挂起服务以便进行基础资源维护。有关使用*冻结*和*解冻*操作的重要信息请参考 “使用冻结和解冻操作的注意事项”一节。	**clusvcadm -Z <service_name>**
「解冻」	解冻会使服务脱离*冻结*状态。这会重新启用状态检查。有关使用*冻结*和*解冻*操作的重要信息请参考 “使用冻结和解冻操作的注意事项”一节。	**clusvcadm -U <service_name>**
「迁移」	将虚拟机迁移到另一个节点中。您必须指定目标节点。根据失败的情况，迁移失败可能导致虚拟机处于*失败*状态，或者在最初拥有者中处于启动的状态。	**clusvcadm -M <service_name> -m <member>** **重要** 在*迁移*操作中您*必须*使用 **-m <member>** 选项指定目标节点。
「重启」	在当前运行该服务的节点中重启服务。	**clusvcadm -R <service_name>**

使用冻结和解冻操作的注意事项

使用*冻结*操作可维护部分 **rgmanager** 服务。例如：如果您有一个数据库和一个网页服务器使用 **rgmanager** 服务，您可能要冻结 **rgmanager** 服务，停止数据库，执行维护，重启数据库，并解冻该服务。

当冻结某个服务时，它会有以下动作：

- 禁用*状态*检查。
- 禁用*启动*操作。
- 禁用*停止*操作。
- 不会出现故障切换（即使关闭该服务的拥有者）。

重要

不按照以下步骤执行将导致将在多台主机中重新分配资源：

- 在冻结服务时您*一定不能*停止 rgmanager 的所有事务，除非您要在重启 rgmanager 前重启该主机。
- 在报告的拥有者重新加入集群并重启 rgmanager 前，您*一定不能*解冻服务。

8.4. 更新配置

更新集群配置包括编辑集群配置文件（**/etc/cluster/cluster.conf**），并在集群的每个节点中推广该文件。您可以使用以下方法之一更新配置：

- 第 8.4.1 节 “使用 **cman_tool version -r** 更新配置”
- 第 8.4.2 节 “使用 **scp** 更新配置”

8.4.1. 使用 cman_tool version -r 更新配置

要使用 **cman_tool version -r** 命令更新配置，请执行以下步骤：

1. 在集群的任意节点中编辑 **/etc/cluster/cluster.conf** 文件。
2. 增加 **config_version** 属性参数即可更新该参数（例如：从 **config_version="2"** 改为 **config_version="3">**）。
3. 保存 **/etc/cluster/cluster.conf**。
4. 运行 **cman_tool version -r** 命令，在其他集群节点中推广该配置。需要在每个集群节点中都运行 **ricci** 方可推广更新的集群配置信息。
5. 确认推广了更新的文件。
6. 如果只更改以下配置，则可以跳过这一步（重启集群软件）：
   - 从集群配置中删除节点 — *除非*节点计数从大于 2 变为等于 2。有关从集群中删除节点，并从两个以上节点变为两个节点的详情请参考 第 8.2 节 “删除或者添加节点”。
   - 在集群配置中添加节点 — *除非*节点计数从 2 变为大于 2。有关在集群中添加节点且从两个节点增加到两个以上节点的详情请参考 第 8.2.2 节 “在集群中添加节点”。
   - 更改记录守护进程信息的方法。
   - HA 服务/VM 维护（添加、编辑或者删除）。
   - 资源维护（添加、编辑或者删除）。
   - 故障切换域维护（添加、编辑或者删除）。

   否则，您必须重启该集群软件，如下：

   1. 在每个节点中如 第 8.1.2 节 “停止集群软件” 所述停止集群软件。例如：

```
[root@example-01 ~]# service rgmanager stop
Stopping Cluster Service Manager:                          [  OK
]
[root@example-01 ~]# service gfs2 stop
Unmounting GFS2 filesystem (/mnt/gfsA):                    [  OK
]
Unmounting GFS2 filesystem (/mnt/gfsB):                    [  OK
]
[root@example-01 ~]# service clvmd stop
Signaling clvmd to exit                                    [
OK  ]
clvmd terminated                                           [
OK  ]
```

```
[root@example-01 ~]# service cman stop
Stopping cluster:
   Leaving fence domain...                                 [
OK  ]
   Stopping gfs_controld...                                [
OK  ]
   Stopping dlm_controld...                                [
OK  ]
   Stopping fenced...                                      [
OK  ]
   Stopping cman...                                        [
OK  ]
   Waiting for corosync to shutdown:                       [  OK
]
   Unloading kernel modules...                             [
OK  ]
   Unmounting configfs...                                  [
OK  ]
[root@example-01 ~]#
```

2. 在每个节点中如 第 8.1.1 节 “启动集群软件” 所述启动集群软件。例如：

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                             [
OK  ]
   Global setup...                                         [
OK  ]
   Loading kernel modules...                               [
OK  ]
   Mounting configfs...                                    [
OK  ]
   Starting cman...                                        [
OK  ]
   Waiting for quorum...                                   [
OK  ]
   Starting fenced...                                      [
OK  ]
   Starting dlm_controld...                                [
OK  ]
   Starting gfs_controld...                                [
OK  ]
   Unfencing self...                                       [
OK  ]
   Joining fence domain...                                 [
OK  ]
[root@example-01 ~]# service clvmd start
Starting clvmd:                                            [
OK  ]
Activating VG(s):   2 logical volume(s) in volume group
"vg_example" now active
                                                           [
OK  ]
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                      [  OK
]
```

```
Mounting GFS2 filesystem (/mnt/gfsB):                      [  OK
]
[root@example-01 ~]# service rgmanager start
Starting Cluster Service Manager:                          [  OK
]
[root@example-01 ~]#
```

停止和启动集群软件可确保所有只在启动时检查的配置更改都包含在运行的配置中。

7. 在任意集群节点中运行 **cman_tool nodes** 以确认那些节点是作为集群的成员运行（在状态列 "Sts" 中被标记为 "M"）。例如：

```
[root@example-01 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M    548   2010-09-28 10:52:21  node-01.example.com
   2   M    548   2010-09-28 10:52:21  node-02.example.com
   3   M    544   2010-09-28 10:52:21  node-03.example.com
```

8. 在任意节点中使用 **clustat** 程序确认那些 HA 服务正常运行。另外 **clustat** 可显示集群节点状态。例如：

```
[root@example-01 ~]#clustat
Cluster Status for mycluster @ Wed Nov 17 05:40:00 2010
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
 node-03.example.com                         3 Online, rgmanager
 node-02.example.com                         2 Online, rgmanager
 node-01.example.com                         1 Online, Local,
rgmanager

 Service Name                   Owner (Last)
State
 ------- ----                   ----- ------                    --
---
 service:example_apache         node-01.example.com
started
 service:example_apache2        (none)
disabled
```

9. 如果集群如预期运行，则您完成了配置更新。

8.4.2. 使用 scp 更新配置

要使用 **scp** 命令更新配置，请执行以下步骤：

1. 在每个节点中如 第 8.1.2 节 “停止集群软件” 所述停止集群软件。例如：

```
[root@example-01 ~]# service rgmanager stop
Stopping Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]# service gfs2 stop
Unmounting GFS2 filesystem (/mnt/gfsA):                    [  OK  ]
Unmounting GFS2 filesystem (/mnt/gfsB):                    [  OK  ]
```

```
[root@example-01 ~]# service clvmd stop
Signaling clvmd to exit                                    [  OK
]
clvmd terminated                                           [  OK
]
[root@example-01 ~]# service cman stop
Stopping cluster:
   Leaving fence domain...                                 [  OK
]
   Stopping gfs_controld...                                [  OK
]
   Stopping dlm_controld...                                [  OK
]
   Stopping fenced...                                      [  OK
]
   Stopping cman...                                        [  OK
]
   Waiting for corosync to shutdown:                       [  OK  ]
   Unloading kernel modules...                             [  OK
]
   Unmounting configfs...                                  [  OK
]
[root@example-01 ~]#
```

2. 在集群的任意节点中编辑 **/etc/cluster/cluster.conf** 文件。

3. 增加 **config_version** 属性参数即可更新该参数（例如：从 **config_version="2"** 改为 **config_version="3">**）。

4. 保存 **/etc/cluster/cluster.conf**。

5. 运行 **ccs_config_validate** 命令，根据集群方案（**cluster.rng**）验证更新的文件。例如：

```
[root@example-01 ~]# ccs_config_validate
Configuration validates
```

6. 如果更新的文件有效，则请使用 **scp** 将其推广到每个集群节点的 **/etc/cluster/** 文件中。

7. 确认推广了更新的文件。

8. 在每个节点中如 第 8.1.1 节 “启动集群软件” 所述启动集群软件。例如：

```
[root@example-01 ~]# service cman start
Starting cluster:
   Checking Network Manager...                             [  OK
]
   Global setup...                                         [  OK
]
   Loading kernel modules...                               [  OK
]
   Mounting configfs...                                    [  OK
]
   Starting cman...                                        [  OK
]
   Waiting for quorum...                                   [  OK
```

```
]
   Starting fenced...                                      [  OK
]
   Starting dlm_controld...                                [  OK
]
   Starting gfs_controld...                                [  OK
]
   Unfencing self...                                       [  OK
]
   Joining fence domain...                                 [  OK
]
[root@example-01 ~]# service clvmd start
Starting clvmd:                                            [  OK
]
Activating VG(s):   2 logical volume(s) in volume group "vg_example"
now active
                                                           [  OK
]
[root@example-01 ~]# service gfs2 start
Mounting GFS2 filesystem (/mnt/gfsA):                      [  OK  ]
Mounting GFS2 filesystem (/mnt/gfsB):                      [  OK  ]
[root@example-01 ~]# service rgmanager start
Starting Cluster Service Manager:                          [  OK  ]
[root@example-01 ~]#
```

9. 在任意集群节点中运行 **cman_tool nodes** 以确认那些节点是作为集群的成员运行（在状态列 "Sts" 中被标记为 "M"）。例如：

```
[root@example-01 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M    548   2010-09-28 10:52:21  node-01.example.com
   2   M    548   2010-09-28 10:52:21  node-02.example.com
   3   M    544   2010-09-28 10:52:21  node-03.example.com
```

10. 在任意节点中使用 **clustat** 程序确认那些 HA 服务正常运行。另外 **clustat** 可显示集群节点状态。例如：

```
[root@example-01 ~]#clustat
Cluster Status for mycluster @ Wed Nov 17 05:40:00 2010
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
 node-03.example.com                         3 Online, rgmanager
 node-02.example.com                         2 Online, rgmanager
 node-01.example.com                         1 Online, Local,
rgmanager

 Service Name                   Owner (Last)
State
 ------- ----                   ----- ------                    --
---
 service:example_apache         node-01.example.com
```

```
started
 service:example_apache2        (none)
disabled
```

11. 如果集群如预期运行，则您完成了配置更新。

第 9 章 诊断并修正集群中的问题

集群问题的故障排除通常比较困难。这是因为由于系统集群复杂性更大，使之与诊断单一系统的问题完全不同。但是有一些问题是管理员在部署或者管理集群时会经常遇到的。了解如何处理那些常见问题可让您更轻松地部署和管理集群。

本章提供有关集群的常见问题以及如何对其进行故障排除。您可以在我们的知识库中获得更多信息，也可联络授权红帽支持代表寻求帮助。如果您的问题是关于 GFS2 文件系统，您可以在 *《全局文件系统 2》* 一书中找到故障排除常见 GFS2 问题的信息。

9.1. 配置更改不生效

修改集群配置后，您必须将那些更改推广到该集群的每个节点中。

- 使用 **Conga** 配置集群时，**Conga** 会在应用那些更改时自动将其推广。
- 有关使用 **ccs** 命令推广集群配置更改的详情请参考 第 5.15 节 “在集群节点中推广配置文件”。
- 有关使用命令行工具推广集群配置更改的详情请参考 第 8.4 节 “更新配置”。

如果在您的集群中进行任何以下配置更改，则在将其推广到集群中后无需重启集群就可使更改生效。

- 从集群配置中删除节点 — *除非* 节点计数由大于 2 变为等于 2。
- 在集群配置中添加节点 — *除非* 节点计数由 2 变为大于 2。
- 更改日志设置。
- 添加、编辑或删除 HA 服务或 VM 组件。
- 添加、编辑或删除集群资源。
- 添加、修改和删除故障切换域。

如果您要更改集群的任何其他配置，就必须重启该集群方可使更改生效。以下集群配置更改需要重启集群后方可生效：

- 在集群配置文件中添加或删除 **two_node** 选项。
- 重命名该集群。
- 更改 **corosync** 或 **openais** 计时器。
- 从仲裁磁盘中添加、更改或删除探试，更改任意仲裁磁盘计时器，或者更改仲裁磁盘设备。要让这些更改生效，则需要在全局重启 **qdiskd** 守护进程。
- 为 **rgmanager** 更改 **central_processing** 模式。要使此更改生效，需要全局重启 **rgmanager**。
- 更改多播地址。
- 将传输方式从 UDP 多播改为 UDP 单播，或者从 UDP 单播改为 UDP 多播。

您可以使用 **Conga**、**ccs** 或者命令行工具重启该集群。

- 有关使用 **Conga** 重启集群的详情请参考 第 4.4 节 “启动、停止、刷新和删除集群”。

- 有关使用 **ccs** 重启集群的详情请参考 第 6.2 节 “启动和停止集群”。
- 有关使用命令行工具重启集群的详情请参考 第 8.1 节 “启动和停止集群软件”。

9.2. 没有形成集群

如果您无法形成新的集群，请检查以下方面：

- 确定正确设置了名称解析。**cluster.conf** 文件中的集群节点名称应该与用来接写通过网络进行沟通的集群地址的名称对应。例如：如果您的集群节点名为 **nodea** 和 **nodeb**，请确定在 **/etc/cluster/cluster.conf** 文件中有这两个节点的条目，且 **/etc/hosts** 文件与那些名称匹配。
- 因为集群使用多播在节点间进行沟通，请确定多播流量没有被阻断、延迟，或者在集群用来沟通的网络中被干扰。请注意：有些 Cisco 开关的功能可能导致多播流量延迟。
- 使用 **telnet** 或者 **SSH** 确认您是否可连接到远程节点。
- 执行 **ethtool eth1 | grep link** 命令检查该以太网链接是否可用。
- 在每个节点中使用 **tcpdump** 命令检查网络流量。
- 确定您没有设定防火墙规则阻断节点间的沟通。
- 确定该集群用于内部节点间沟通的接口没有使用捆绑模式 0、1 和 2 以外的模式。（从红帽企业版 Linux 6.4 开始支持捆绑模式 0 和 2。）

9.3. 无法在 FENCE 或者重启后重新加入集群的节点

如果您从节点在 fence 或者重启后无法重新加入该集群，请检查以下方面：

- 使用 Cisco Catalyst 切换通过其流量的集群可能会有这个问题。
- 请确定所有集群节点都使用同一版本的 **cluster.conf** 文件。如果在任何一个节点中的 **cluster.conf** 文件有所不同，则那些节点在 fence 后就无法加入该集群。

  从红帽企业版 Linux 6.1 开始，您可以使用以下命令确认在主机的集群配置文件中指定的所有节点都有相同的集群配置文件：

  ```
  ccs -h host --checkconf
  ```

  有关 **ccs** 命令的详情请参考 *第 5 章 使用 ccs 命令配置红帽高可用性附加组件* 和 *第 6 章 使用 ccs 管理 Red Hat 高可用性附加组件*。
- 请确定您在要加入该集群的节点中为集群服务配置了 **chkconfig on**。
- 请确定没有阻断该节点与集群中的其他节点沟通的防火墙规则。

9.4. 集群守护进程崩溃

RGManager 有一个监控进程可在主 **rgmanager** 进程意外失败时重启该主机。这就可以 fence 该集群节点，且 **rgmanager** 可在另一台主机中恢复该服务。监控守护进程探测到主 **rgmanager** 崩溃时，它就会重启该集群节点，同时活动的集群节点将探测到该集群节点已离开，并将其从该集群中逐出。

较小数字的*进程 ID*（PID）是 watchdog 进程，可在其子进程（有较大 PID 数字的进程）崩溃时起作用。使用 **gcore** 捕获 PID 较大进程的 core 可帮助对崩溃的守护进程进行故障排除。

安装所需软件包捕获和查看 core，并保证 **rgmanager** 和 **rgmanager-debuginfo** 是同一版本，否则捕获的应用程序 core 可能不可用。

```
$ yum -y --enablerepo=rhel-debuginfo install gdb rgmanager-debuginfo
```

9.4.1. 在运行时捕获 rgmanager Core

它启动时有两个 **rgmanager** 进程。您必须捕获有较高 PID 的那个 **rgmanager** 进程。

以下是执行 **ps** 命令时显示两个 **rgmanager** 进程的输出结果示例。

```
$ ps aux | grep rgmanager | grep -v grep

root     22482  0.0  0.5  23544  5136 ?        S<Ls Dec01   0:00 rgmanager
root     22483  0.0  0.2  78372  2060 ?        S<l  Dec01   0:47 rgmanager
```

在下面的示例中，使用 **pidof** 程序自动确定 pid 数字较高的进程，即生成 core 的那个进程的 pid。该命令为进程 22483，就是那个较高的 pid 数字，捕获程序 core。

```
$ gcore -o /tmp/rgmanager-$(date '+%F_%s').core $(pidof -s rgmanager)
```

9.4.2. 守护进程崩溃是捕获 Core

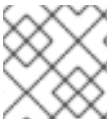
默认情况下 **/etc/init.d/functions** 脚本会阻断由 **/etc/init.d/rgmanager** 所调用守护进程的 core 文件。对于要生成应用程序 core 的守护进程，您必须启用那个选项。必须在所有需要捕获应用程序 core 的集群节点中执行这一步骤。

要在 rgmanager 守护进程崩溃时生成 core 文件，请编辑 **/etc/sysconfig/cluster** 文件。***DAEMONCOREFILELIMIT*** 参数可让该守护进程在该进程崩溃时生成 core 文件。使用 **-w** 选项可阻止 watchdog 进程运行。如果 **rgmanager** 崩溃，watchdog 守护进程负责重启该集群节点。如果 watchdog 守护进程正在运行，则不会生成该 core 文件，因此一定要禁用它方可捕获 core 文件。

```
DAEMONCOREFILELIMIT="unlimited"
RGMGR_OPTS="-w"
```

重启 rgmanager 激活新配置选项：

```
service rgmanager restart
```



注意

如果在这个集群节点中正在运行集群服务，那么它会以非良好状态离开运行的服务。

如果 **rgmanager** 进程崩溃生成 core 文件，那么该文件将可以被写入。

```
ls /core*
```

-

输出结果应类似如下：

```
/core.11926
```

重启 **rgmanager** 捕获应用程序 core 前，请移动或删除 / 目录中的所有旧 core 文件。应重启出现 **rgmanager** 崩溃的集群节点，或者在捕获 core 文件后 fence 该节点以保证没有运行监视进程。

9.4.3. 记录 gdb Backtrace 会话

捕获 core 文件后，您可以使用 **gdb**，即 GNU Debugger 查看其内容。要在受影响系统的 core 文件中记录 **gdb** 脚本会话，请运行以下命令：

```
$ script /tmp/gdb-rgmanager.txt
$ gdb /usr/sbin/rgmanager /tmp/rgmanager-.core.
```

这样将启动 **gdb** 会话，同时 **script** 会将其记录到适当的文本文件中。同时在 **gdb** 中运行以下命令：

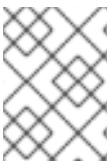
```
(gdb) thread apply all bt full
(gdb) quit
```

按 **ctrl-D** 停止脚本会话，并将其保存到文本文件中。

9.5. 集群服务挂起

当集群服务尝试 fence 某个节点时，该集群服务会停止，直到成功完成 fence 操作。因此，如果您使用集群控制的存储或者服务挂起，且集群节点显示不同的集群成员，或者当您尝试 fence 某个节点时集群挂起，您需要重启节点进行恢复时，请检查以下方面：

- 该集群可能尝试 fence 某个节点，且 fence 操作可能已经失败。
- 查看所有节点中的 **/var/log/messages** 文件，看看是否有失败的 fence 信息。如果有，重启集群中的那些节点，并正确配置 fencing。
- 确认没有出现如 第 9.8 节 “双节点集群的每个节点都报告第二个节点无法工作” 所示的网络分割。同时确认节点间可进行沟通，网络正常工作。
- 如果有节点离开该集群，剩余的节点可能不足构成集群。集群需要有一定量的节点方可操作。如果删除节点导致该节点没有足够量的节点，则服务和存储将会挂起。您可以调整预期的票数或者在该集群中保持所需节点数。



注意

您可以使用 **fence_node** 命令或者 **Conga** 手动 fence 某个节点。详情请查看 **fence_node** man page 和 第 4.3.2 节 “使节点离开或者加入集群”。

9.6. 无法启动集群服务

如果无法启动某个集群控制的服务，请检查以下方面。

- 可能在 **cluster.conf** 文件的服务配置部分有语法错误。您可以使用 **rg_test** 命令验证您配置文件中的语法。如果有任何配置或者语法错误，**rg_test** 会告诉您哪里出了问题。

```
$ rg_test test /etc/cluster/cluster.conf start service servicename
```

有关 **rg_test** 命令的详情请参考 第 C.5 节 “调整并测试服务和资源顺序”。

如果配置无误，可提高资源组管理器的日志级别，然后阅读日志信息，确定是什么导致无法启动该服务。您可以在 **cluster.conf** 文件的 **rm** 标签中添加 **loglevel="7"** 参数提高日志级别。然后您可以根据启动、停止、迁移集群的服务增加信息日志的详细程度。

9.7. 无法迁移集群控制的服务

如果无法将集群控制的服务转移到另一个节点，但可在某些具体节点中启动，请检查以下方面。

- 请确定该集群的所有需要运行给定服务的节点中都有运行那个服务需要的资源。例如：如果您的集群服务假设某个脚本文件位于某具体位置，或者某个文件系统挂载于某个具体挂载点，那么您必须确定那些资源位于该集群的所有节点的预期位置。
- 请确定没有将故障切换域、服务相依性以及服务专有权配置为您无法将服务迁移到您需要的节点中。
- 如果所需服务是一个虚拟机资源，请查看文档以确定完成了所有正确配置工作。
- 如 第 9.6 节 “无法启动集群服务” 所述提高资源组管理器日志级别，然后阅读信息日志以确定是什么导致无法迁移该服务。

9.8. 双节点集群的每个节点都报告第二个节点无法工作

如果您的集群是一个双节点集群，且每个节点都报告它可工作，但另一个无法工作，这说明您的集群节点无法通过集群心跳网络（heartbeat network）的多播进行沟通。我们称之为“脑裂（split brain）”或者“网络分割（network partition）”。要解决这个问题，请检查 第 9.2 节 “没有形成集群” 中所述的情况。

9.9. 在 LUN 路径失败中 FENCE 的节点

如果无论何时您有 LUN 路径失败时，您集群中的一个或者多个节点被 fence，这可能是优先使用仲裁磁盘而没有使用多路径存储造成的。如果您使用仲裁磁盘，且您的仲裁磁盘优先于多路径存储，请确定您正确配置了所有承受路径失败的时限。

9.10. 仲裁磁盘不作为集群成员出现

如果您将系统配置为使用仲裁磁盘，但该仲裁磁盘没有作为您集群的成员出现，请检查以下情况。

- 请确定您为 **qdisk** 服务设定了 **chkconfig on**。
- 请确定您启动了 **qdisk** 服务。
- 注：在该集群中注册仲裁磁盘可能需要几分钟，这很正常。

9.11. 异常故障切换行为

集群服务器的常见问题之一是异常故障切换行为。当启动其它服务时服务会停止，或者服务在故障切换时拒绝启动。这可能是由于故障切换域、服务相依性以及服务排他性造成故障切换系统比较复杂。尝试使用简单一些的服务或者故障切换域配置，看看问题是否还存在。避免使用类似服务排他性和相依性功能，除非您完全掌握这些功能在所有情况下对故障切换造成的影响。

9.12. 随机发生 FENCING

如果您发现某个节点随机发生 fence，请检查以下情况。

- 造成 fence 的根本原因*总是*因为某个节点丢失 token，就是说它无法与集群中的其他节点沟通，并停止返回心跳。
- 任何情况下，如果某个系统不在指定的 token 间隔中返回心跳都会导致出现 fence。默认情况下 token 间隔为 10 秒。您可以在 **cluster.conf** 文件 totem 标签的 tokem 参数中以毫秒为单位将其指定为所需值（例如：设定为 **totem token="30000"** 即为 30 秒）。
- 请确定网络可正常工作。
- 确定该集群用于内部节点间沟通的接口没有使用捆绑模式 0、1 和 2 以外的模式。（从红帽企业版 Linux 6.4 开始支持捆绑模式 0 和 2。）
- 设法确定该系统是否为 "freezing" 或者内核 panicking。设置 **kdump** 程序看看您是否可以从这些 fence 之一中得到一个 core。
- 确定不会出现错误地归咎于某个 fence 的情况，例如因为某个存储失败弹出仲裁磁盘，或者由于一些外部因素造成类似 Oracle RAC 的第三方产品重启某个节点。这些信息日志通常对确定这样的问题非常有帮助。无论何时 fence 节点或者重启节点，标准的操作都应该是查看从发生重启/fence 时起该集群中所有节点的信息日志。
- 对于可能导致系统如预期响应心跳的硬件错误要彻底检查系统。

9.13. 需启用发布式锁定管理器（DLM）的 DEBUG 日志

如果需要，您可为发布式锁定管理器（DLM）启用两个 debug 选项：即 DLM 内核 debugging，和 POSIX 锁定 debugging。

要启用 DLM debugging，请编辑 **/etc/cluster/cluster.conf** 文件，在 **dlm** 标签中添加配置选项。**log_debug** 选项启用 DLM 内核 debugging 信息，**plock_debug** 选项启用 POSIX 锁定 debugging 信息。

以下 **/etc/cluster/cluster.conf** 的示例部分演示了启用 DLM debug 选项的 **dlm** 标签：

```
<cluster config_version="42" name="cluster1">
  ...
  <dlm log_debug="1" plock_debug="1"/>
  ...
</cluster>
```

编辑 **/etc/cluster/cluster.conf** 文件后，请运行 **cman_tool version -r** 命令在其它集群节点中推广该配置。

第 10 章 使用红帽高可用性附加组件进行 SNMP 配置

从红帽企业版 Linux 6.1 开始，红帽高可用性附加组件支持 SNMP 陷阱。本章论述了如何为 SNMP 配置您的系统，并附带红帽高可用性附加组件为具体集群事件使用的陷阱小结。

10.1. SNMP 和红帽高可用性附加组件

红帽高可用性附加组件 SNMP 的子代理为 **foghorn**，它可发出 SNMP 陷阱。**foghorn** 代理与 **snmpd** 守护进程通过 AgentX 协议进行对话。**foghorn** 子代理只生成 SNMP 陷阱，不支持其它 SNMP 操作，比如 **get** 或者 **set**。

目前 **foghorn** 子代理没有 **config** 选项。无法将其配置为使用具体插槽，目前只支持默认的 AgentX 插槽。

10.2. 使用红帽高可用性附加组件配置 SNMP

要使用红帽高可用性附加组件配置 SNMP，请在该集群的每个节点中执行以下步骤，保证启用并运行了所需服务。

1. 要在红帽高可用性附加组件中使用 SNMP 陷阱，则要求将 **snmpd** 作为主代理运行。因为 **foghorn** 服务是子代理，且使用 AgentX 协议，您必须在 **/etc/snmp/snmpd.conf** 文件中添加以下行启用 AgentX 支持：

   ```
   master agentx
   ```

2. 要指定发送 SNMP 陷阱通知的主机，请在 **/etc/snmp/snmpd.conf** 文件中添加以下行：

   ```
   trap2sink host
   ```

   有关通知处理的详情请参考 **snmpd.conf** man page。

3. 执行以下命令确定启用并运行了 **snmpd** 守护进程：

   ```
   # chkconfig snmpd on
   # service snmpd start
   ```

4. 如果没有启用并运行 **messagebus**，请执行以下命令：

   ```
   # chkconfig messagebus on
   # service messagebus start
   ```

5. 执行以下命令确定启用并运行 **foghorn** 守护进程：

   ```
   # chkconfig foghorn on
   # service foghorn start
   ```

6. 执行以下命令配置您的系统以便 **COROSYNC-MIB** 生成 SNMP 陷阱，并确定启用并运行 **corosync-notifyd** 守护进程：

```
# echo "OPTIONS=\"-d\" " > /etc/sysconfig/corosync-notifyd
# chkconfig corosync-notifyd on
# service corosync-notifyd start
```

您在集群的每个节点中配置 SNMP 并确定所需服务都在运行后，**foghorn** 服务会受到 D-bus 信号，并将其转换为 SNMPv2 陷阱。会将这些陷阱发送到使用 **trapsink** 条目定义的主机中以便接收 SNMPv2 陷阱。

10.3. 转发 SNMP 陷阱

可以将 SNMP 陷阱转发到不属于该集群的机器中，您可在外部机器中使用 **snmptrapd** 守护进程并自定义如何响应通知。

执行以下步骤在集群中将 SNMP 陷阱转发到不是该集群节点的机器中：

1. 在该集群的每个节点中如 第 10.2 节 “使用红帽高可用性附加组件配置 SNMP” 所述执行以下步骤，在 **/etc/snmp/snmpd.conf** 文件中设置 **trap2sink *host*** 条目，指定将要运行 **snmptrapd** 守护进程的外部主机。

2. 在将要接收陷阱的外部主机中编辑 **/etc/snmp/snmptrapd.conf** 配置文件，指定您的社区字符串。例如：使用以下条目可让 **snmptrapd** 守护进程使用 **public** 社区字符串处理通知。

   ```
   authCommunity log,execute,net public
   ```

3. 执行以下命令在接收陷阱的外部主机中确定启用并运行 **snmptrapd** 守护进程：

   ```
   # chkconfig snmptrapd on
   # service snmptrapd start
   ```

有关处理 SNMP 通知的详情请参考 **snmptrapd.conf** man page。

10.4. 红帽高可用性附加组件产生的 SNMP 陷阱

foghorn 守护进程生成以下陷阱：

- **fenceNotifyFenceNode**

  在被 fence 的节点尝试 fence 另一个节点时会出现这个陷阱。注：只会在一个节点中生成这个陷阱 -- 即在尝试执行 fence 操作的节点中生成。这个通知包括以下字段：

  - **fenceNodeName** -- 被 fence 的节点名称
  - **fenceNodeID** -- 被 fence 的节点 id
  - **fenceResult** -- fence 操作结果（0 表示成功，-1 表示有问题，-2 表示没有定义 fencing 方法）

- **rgmanagerServiceStateChange**

  集群服务状态更改时会出现这个陷阱。该通知包括以下字段：

  - **rgmanagerServiceName** -- 该服务名称，其中包括服务类型（例如：**service:foo** 或者 **vm:foo**）。

- **rgmanagerServiceState** -- 该服务的状态。这不包括过渡状态，比如 **starting** 和 **stopping**，以减小陷阱中的杂乱程度。
- **rgmanagerServiceFlags** -- 服务标签。以下是目前支持的两个标签：**frozen** 表示已经使用 **clusvcadm -Z** 冻结服务；**partial** 表示在该服务中将失败的资源标记为 **non-critical**，这样可在该资源失败并手动重启其组件时不影响整个服务。
- **rgmanagerServiceCurrentOwner** -- 服务拥有者。如果该服务没有运行，则该字段为 **(none)**。
- **rgmanagerServicePreviousOwner** -- 如果知道，则列出最后的服务拥有者。如果不知道最后的拥有者，该字段会显示 **(none)**。

corosync-nodifyd 守护进程生成以下陷阱：

- **corosyncNoticesNodeStatus**

  当节点加入或者离开集群时会出现这个陷阱。该通知包括以下字段：

  - **corosyncObjectsNodeName** -- 节点名称
  - **corosyncObjectsNodeID** -- 节点 id
  - **corosyncObjectsNodeAddress** -- 节点 IP 地址
  - **corosyncObjectsNodeStatus** -- 节点状态（**joined** 或者 **left**）

- **corosyncNoticesQuorumStatus**

  仲裁状态更改时会出现这个陷阱。该通知包括以下字段：

  - **corosyncObjectsNodeName** -- 节点名称
  - **corosyncObjectsNodeID** -- 节点 id
  - **corosyncObjectsQuorumStatus** -- 仲裁的新状态（**quorate** 或者 **NOT quorate**）

- **corosyncNoticesAppStatus**

  客户端程序连接到 Corosync 或者与之断开连接时会出现这个陷阱。

  - **corosyncObjectsNodeName** -- 节点名称
  - **corosyncObjectsNodeID** -- 节点 id
  - **corosyncObjectsAppName** -- 应用程序名称
  - **corosyncObjectsAppStatus** -- 该应用程序的新状态（**connected** 或者 **disconnected**）

第 11 章 集群 SAMBA 配置

从红帽企业版 Linux 6.2 发行本开始，红帽高可用附加组件为配置为 active/active 的运行中的集群 Samba 提供支持。中要求您在集群的所有节点中安装并配置 CTDB，与 GFS2 集群的系统文件一同使用。

注意

红帽企业版 Linux 6 最多支持 4 个运行集群 Samba 的节点。

本章通过配置示例系统论述配置 CTDB 的步骤。有关配置 GFS2 文件系统的详情请参考 *《全局文件系统 2》*。有关配置逻辑卷的详情请参考 *《管理逻辑卷管理器》*。

11.1. CTDB 概述

CTDB 是 Samba 使用的 TDB 数据库的集群实施。要使用 CTDB，则必须有一个可用的集群文件系统，且该文件系统可为该集群中的所有节点共享。CTDB 在这个集群的文件系统顶层提供集群的功能。从红帽企业版 Linux 6.2 开始，CTDB 还可运行与红帽企业版 Linux 集群提供的堆栈平行的堆栈。CTDB 管理节点成员、执行恢复/故障切换、IP 重新定位以及 Samba 服务。

11.2. 所需软件包

除运行红帽高可用附加组件以及红帽弹性存储附加组件所需标准软件包外，使用红帽企业版 Linux 集群运行 Samba 还需要以下软件包：

- **ctdb**
- **samba**
- **samba-common**
- **samba-winbind-clients**

11.3. GFS2 配置

使用红帽企业版 Linux 集群配置 Samba 需要两个 GFS2 文件系统：一个小文件系统用于 CTDB，第二个文件系统用于 Samba 共享。这个示例演示了如何生成两个 GFS2 文件系统。

在生成 GFS2 文件系统前，请首先为每个文件系统生成 LVM 逻辑卷。有关生成 LVM 逻辑卷的详情，请参考 *《管理逻辑卷管理器》*。这个示例使用以下两个逻辑卷：

- **/dev/csmb_vg/csmb_lv**，保存将要使用 Samba 共享导出的用户数据，并根据情况确定大小。这个示例生成大小为 100GB 的逻辑卷。
- **/dev/csmb_vg/ctdb_lv**，保存共享 CTDB 状态信息，大小应为 1GB.

您只能在集群的一个节点中生成集群的卷组和逻辑卷。

要在逻辑卷中生成 GFS2，请运行 **mkfs.gfs2** 命令。您只能在一个集群节点中运行这个命令。

要在逻辑卷 **/dev/csmb_vg/csmb_lv** 中生成托管 Samba 共享的文件系统，请执行以下命令：

```
[root@clusmb-01 ~]# mkfs.gfs2 -j3 -p lock_dlm -t csmb:gfs2
/dev/csmb_vg/csmb_lv
```

参数含义如下：

-j

指定在该文件系统中生成的日志数。这个示例使用有三个节点的集群，因此我们为每个节点生成一个日志。

-p

指定锁定协议。**lock_dlm** 是 GFS2 用来在节点间进行沟通的锁定协议。

-t

指定锁定表名称，格式为 *cluster_name:fs_name*。在这个示例中，在 **cluster.conf** 文件中指定的集群名称为 **csmb**，同时我们使用 **gfs2** 作为该文件系统名称。

这个命令的输出结果如下：

```
This will destroy any data on /dev/csmb_vg/csmb_lv.
  It appears to contain a gfs2 filesystem.

Are you sure you want to proceed? [y/n] y

Device:
/dev/csmb_vg/csmb_lv
Blocksize:  4096
Device Size  100.00 GB (26214400 blocks)
Filesystem Size: 100.00 GB (26214398 blocks)
Journals:  3
Resource Groups:  400
Locking Protocol:   "lock_dlm"
Lock Table:   "csmb:gfs2"
UUID:
  94297529-ABG3-7285-4B19-182F4F2DF2D7
```

在这个示例中，会在所有节点的 **/mnt/gfs2** 中挂载 **/dev/csmb_vg/csmb_lv** 文件系统。这个挂载点必须与您在 **/etc/samba/smb.conf** 文件 **path =** 选项中指定的 **share** 目录位置值匹配，如 第 11.5 节 “Samba 配置” 所述。

要在逻辑卷 **/dev/csmb_vg/ctdb_lv** 中生成托管 CTDB 状态信息的文件系统，请执行以下命令：

```
[root@clusmb-01 ~]# mkfs.gfs2 -j3 -p lock_dlm -t csmb:ctdb_state
/dev/csmb_vg/ctdb_lv
```

注：这个命令指定的锁定表名称与在 **/dev/csmb_vg/csmb_lv** 中生成文件系统示例中指定的锁定表名称不同。这样可区别在该文件系统中不同设备使用的锁定表名称。

mkfs.gfs2 命令输出结果如下：

```
This will destroy any data on /dev/csmb_vg/ctdb_lv.
  It appears to contain a gfs2 filesystem.

Are you sure you want to proceed? [y/n] y

Device:
```

```
/dev/csmb_vg/ctdb_lv
Blocksize:    4096
Device Size   1.00 GB (262144 blocks)
Filesystem Size:  1.00 GB (262142 blocks)
Journals:  3
Resource Groups:  4
Locking Protocol:  "lock_dlm"
Lock Table:   "csmb:ctdb_state"
UUID:
  BCDA8025-CAF3-85BB-B062-CC0AB8849A03
```

在这个示例中，会在所有节点的 **/mnt/gfs2** 中挂载 **/dev/csmb_vg/ctdb_lv** 文件系统。这个挂载点必须与您在 **/etc/sysconfig/ctdb** 文件 **CTDB_RECOVERY_LOCK** 选项中指定的 **.ctdb.lock** 文件位置匹配，如 第 11.4 节 “CTDB 配置” 所述。

11.4. CTDB 配置

CTDB 配置文件位于 **/etc/sysconfig/ctdb**。以下是必须为 CTDB 操作配置的字段：

- **CTDB_NODES**
- **CTDB_PUBLIC_ADDRESSES**
- **CTDB_RECOVERY_LOCK**
- **CTDB_MANAGES_SAMBA**（必须启用）
- **CTDB_MANAGES_WINBIND**（如果在成员服务器中云系则必须启用）

以下示例演示了使用示例参数为 CTDB 操作设置强制字段的配置文件：

```
CTDB_NODES=/etc/ctdb/nodes
CTDB_PUBLIC_ADDRESSES=/etc/ctdb/public_addresses
CTDB_RECOVERY_LOCK="/mnt/ctdb/.ctdb.lock"
CTDB_MANAGES_SAMBA=yes
CTDB_MANAGES_WINBIND=yes
```

这些参数的含义如下：

CTDB_NODES

指定包含该集群节点列表文件的位置。

CTDB_NODES 参考只列出该集群节点 IP 地址的 **/etc/ctdb/nodes** 文件，例如：

```
192.168.1.151
192.168.1.152
192.168.1.153
```

在这个示例中，每个节点中只有一个接口/IP 可用于集群/CTDB 沟通并为客户端提供服务。但强烈建议每个集群节点都有两个网络接口，这样一个接口设置可专门用于集群/CTDB 沟通，而另一个接口设置可专门用于公用客户端访问。在此使用正确的集群网络 IP 地址，并保证在 **cluster.conf** 文件中使用同一主机名/IP 地址。同样，在 **public_addresses** 文件中为客户端访问使用正确的公共网络接口。

/etc/ctdb/nodes 文件在所有节点中的一致性至关重要，因为顺序很重要，同时如果 CTDB 在不同节点中找到的信息不同就会失败。

CTDB_PUBLIC_ADDRESSES

指定列出用来访问由这个集群导出的 Samba 共享的 IP 地址的文件位置。这些是您要在 DNS 中为集群的 Samba 服务器名称配置的 IP 地址，也是 CIFS 客户端将要连接的地址。将集群 Samba 服务器名称配置为有多个 IP 地址的 DNS 类型 A 记录，并在该集群的客户端中发布轮询 DNS 。

在这个示例中，我们在所有 **/etc/ctdb/public_addresses** 文件列出的地址中配置轮询 DNS 条目 **csmb-server**。DNS 将发布那些在集群中以轮询方式使用这个条目的客户端。

每个节点中 **/etc/ctdb/public_addresses** 文件的内容如下：

```
192.168.1.201/0 eth0
192.168.1.202/0 eth0
192.168.1.203/0 eth0
```

这个示例使用目前在该网络中使用的三个地址。在您自己的配置中，请选择预期客户端可访问的地址。

另外，这个示例显示了集群中 **/etc/ctdb/public_addresses** 文件的内容，其中三个节点，但却有四个公共地址。在这个示例中，IP 地址 198.162.2.1 可由节点 0 和节点 1 托管，只要有一个节点可用，客户端就可以访问这个地址。只有节点 0 和节点 1 都失败的时候，客户端才不能访问这个公共地址。所有其他公共地址只能分别由一个节点提供，因此如果只有在相应的节点可用时该公共地址方可用。

节点 0 中的 **/etc/ctdb/public_addresses** 文件包含以下内容：

```
198.162.1.1/24 eth0
198.162.2.1/24 eth1
```

节点 1 中的 **/etc/ctdb/public_addresses** 文件包含以下内容：

```
198.162.2.1/24 eth1
198.162.3.1/24 eth2
```

节点 2 中的 **/etc/ctdb/public_addresses** 文件包含以下内容：

```
198.162.3.2/24 eth2
```

CTDB_RECOVERY_LOCK

指定 CTDB 内部用来恢复的锁定文件。这个文件必须位于共享存储中，这样所有集群节点都可访问。本小节中的示例使用 GFS2 文件系统，该文件系统会挂载于所有节点的 **/mnt/ctdb**。这与将要导出 Samba 共享的 GFS2 文件系统不同。这个恢复锁定文件的目的是防止出现裂脑（split-brain）。使用 CTDB 较新的版本（1.0.112 及之后的版本）时，可自选是否指定这个文件，只要有防止裂脑的机制即可。

CTDB_MANAGES_SAMBA

当将其设定为 **yes** 启用它时，如果需要提供服务迁移/故障切换，则指定允许 CTDB 启动和停止 Samba 服务。

启用 **CTDB_MANAGES_SAMBA** 时，应禁用 **smb** 和 **nmb** 守护进程的自动 **init** 启动，方法为执行以下命令：

```
[root@clusmb-01 ~]# chkconfig snb off
[root@clusmb-01 ~]# chkconfig nmb off
```

CTDB_MANAGES_WINBIND

当将其设定为 **yes** 启用它时，则指定 CTDB 可根据需要启动和停止 **winbind** 守护进程。当您在 Windows 域或在 active directory 安全模式中使用 CTDB 时应该启用它。

启用 **CTDB_MANAGES_WINBIND** 时，应禁用 **winbind** 守护进程的自动 **init** 启动，方法为执行以下命令：

```
[root@clusmb-01 ~]# chkconfig windinbd off
```

11.5. SAMBA 配置

在这个示例中，Samba 配置文件 **smb.conf** 位于 **/etc/samba/smb.conf**。它包含以下参数：

```
[global]
 guest ok = yes
 clustering = yes
 netbios name = csmb-server
[csmb]
 comment = Clustered Samba
  public = yes
 path = /mnt/gfs2/share
 writeable = yes
 ea support = yes
```

这个示例使用名称 **csmb** 导出位于 **/mnt/gfs2/share** 的共享。这与 **/mnt/ctdb/.ctdb.lock** 中的 GFS2 共享文件系统不同，我们将后者在 **/etc/sysconfig/ctdb** CTDB 配置文件中指定为 **CTDB_RECOVERY_LOCK** 参数。

在这个示例中，我们在首次挂载它时在 **/mnt/gfs2** 中生成 **share** 目录中。**clustering = yes** 条目让 Samba 使用 CTDB。**netbios name = csmb-server** 条目明确设置所有节点有通用的 NetBIOS 名称。如果您计划使用扩展属性，则允许要 **ea support** 参数。

smb.conf 配置文件必须在所有集群节点中是一致的。

Samba 还可使用 **net conf** 命令提供基于注册的配置，自动在集群成员间同步，而无需在集群节点间手动复制配置文件。有关 **net conf** 命令的详情请参考 **net**(8) man page。

11.6. 启动 CTDB 和 SAMBA 服务

启动该集群后，必须挂载您生成的 GFS2 文件系统，如 第 11.3 节 “GFS2 配置” 所述。应为客户端访问设置 Samba **share** 目录权限和集群节点中的用户帐户。

在所有节点中执行以下命令启动 **ctdbd** 守护进程。因为这个示例是使用 **CTDB_MANAGES_SAMBA=yes** 配置 CTDB，所以 CTDB 还在所有节点中启动 Samba 服务，并导出所有配置的 Samba 共享。

```
[root@clusmb-01 ~]# service ctdb start
```

CTDB 启动 Samba 、导出共享并稳定需要几分钟。执行 **ctdb status** 可显示 CTDB 状态，如以下示例所示：

```
[root@clusmb-01 ~]# ctdb status
Number of nodes:3
pnn:0 192.168.1.151     OK (THIS NODE)
pnn:1 192.168.1.152     OK
pnn:2 192.168.1.153     OK
Generation:1410259202
Size:3
hash:0 lmaster:0
hash:1 lmaster:1
hash:2 lmaster:2
Recovery mode:NORMAL (0)
Recovery master:0
```

当您看到所有节点都“OK”后，就可以安全使用集群的 Samba 服务器，如 第 11.7 节 “使用集群的 Samba 服务器” 所述。

11.7. 使用集群的 SAMBA 服务器

客户端可以连接到 Samba 共享，这些共享是通过连接到在 **/etc/ctdb/public_addresses** 文件中指定的 IP 地址之一导出的，也可以是使用我们之前配置的 **csmb-server** DNS 条目导出，如下所示：

```
[root@clusmb-01 ~]# mount -t cifs //csmb-server/csmb /mnt/sambashare -o
user=testmonkey
```

或者

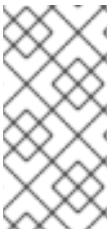
```
[user@clusmb-01 ~]$ smbclient //csmb-server/csmb
```

附录 A. FENCE 设备参数

本附录提供 fence 设备参数描述表。您可以使用 **luci** 配置参数，方法是使用 **ccs** 命令，或编辑 **etc/cluster/cluster.conf** 文件。有关每个 fence 代理的 fence 设备参数完整列表及描述，请参考该代理的 man page。

注意

使用 fence 设备的「**Name**」参数为红帽高可用性附加组件使用的设备指定任意名称。这与该设备的 DNS 名称不同。



注意

某些 fence 设备有可选的「**Password Script**」参数。「**Password Script**」参数可让您使用脚本而不是「**Password**」参数提供 fence 设备密码。使用「**Password Script**」参数可取代「**Password**」参数，允许在集群配置文件（**/etc/cluster/cluster.conf**）不显示密码。

表 A.1 “Fence 设备小结” 列出 fence 设备、与该 fence 设备关联的 fence 设备代理，并提供该 fence 设备参数文档。

表 A.1. Fence 设备小结

Fence 设备	Fence 代理	参数描述参考
APC 电源开关（telnet/SSH）	fence_apc	表 A.2 “APC 电源开关（telnet/SSH）”
Brocade 光纤开关	fence_brocade	表 A.4 “Brocade 光纤开关”
Cisco MDS	fence_cisco_mds	表 A.5 “Cisco MDS”
Cisco UCS	fence_cisco_ucs	表 A.6 “Cisco UCS”
Dell DRAC 5	fence_drac5	表 A.7 “Dell DRAC 5”
Eaton 网络电源控制器（SNMP 接口）	fence_eaton_snmp	表 A.8 “Eaton 网络电源控制器（SNMP 接口）（红帽企业版 Linux 6.4 及之后的版本）”
Egenera SAN 控制器	fence_egenera	表 A.9 “Egenera SAN 控制器”
ePowerSwitch	fence_eps	表 A.10 “ePowerSwitch”
Fence virt	fence_virt	表 A.11 “Fence virt”

Fence 设备	Fence 代理	参数描述参考
富士通-西门子远程查看服务栏（RSB）	fence_rsb	表 A.12 “富士通-西门子远程查看服务栏（RSB）”
惠普刀片机系统	fence_hpblade	表 A.13 “HP 刀片机系统（红帽企业版 Linux 6.4 及之后的版本）”
HP iLO/iLO2 (Integrated Lights Out)	fence_ilo	表 A.14 “HP iLO/iLO2 (Integrated Lights Out)”
惠普 iLO（Integrated Lights Out）MP	fence_ilo_mp	表 A.15 “惠普 iLO（Integrated Lights Out）MP”
IBM 刀片服务器	fence_bladecenter	表 A.16 “IBM 刀片服务器”
IBM 刀片服务器 SNMP	fence_ibmblade	表 A.17 “IBM 刀片服务器 SNMP”
IBM iPDU	fence_ipdu	表 A.18 “IBM iPDU（红帽企业版 Linux 6.4 及之后的版本）”
IF MIB	fence_ifmib	表 A.19 “IF MIB”
Intel 模块化	fence_intelmodular	表 A.20 “Intel 模块化”
IPMI（智能平台管理界面）LAN	fence_ipmilan	表 A.21 “IPMI（智能平台管理界面）LAN”
RHEV-M REST API	fence_rhevm	表 A.22 “RHEV-M REST API（RHEL 6.2 及之后的版本，RHEV 3.0 及之后的版本）”
SCSI Fencing	fence_scsi	表 A.23 “SCSI Fencing”
VMware Fencing（SOAP 接口）	fence_vmware_soap	表 A.24 “VMware Fencing（SOAP 接口）（红帽企业版 Linux 6.2 及之后的版本）”
WTI 电源开关	fence_wti	表 A.25 “WTI 电源开关”

表 A.2 “APC 电源开关（telnet/SSH）” 列出 **fence_apc** 使用的 fence 设备参数，APC 在 telnet/SSH 中使用的 fence 代理。

表 A.2. APC 电源开关（telnet/SSH）

luci 字段	cluster.conf 属性	描述
名称	**name**	连接到集群的 APC 设备名称，在该设备中记录使用 telent/ssh 的 fence 守护进程日志。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址或者主机名。
IP 端口（可选）	**ipport**	用来连接到该设备的 TCP 端口。
登录	**login**	访问该设备的登录名称。
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
电源等待	**power_wait**	执行 power off 或者 power on 命令后要等待的秒数。
端口	**port**	该端口
开关（可选）	**switch**	当您使用多个菊花链（daisy-chained）开关时连接到该节点的 APC 开关的开关数。
使用 SSH	**secure**	表示系统将使用 SSH 访问该设备。
到 SSH 识别文件的路径	**identity_file**	SSH 的识别文件。

表 A.3 “使用 SNMP 的 APC 电源开关” 列出 **fence_apc_snmp** 使用的 fence 设备参数；通过 SNMP 协议登录到 SNP 设备的 APC fence 代理。

表 A.3. 使用 SNMP 的 APC 电源开关

luci 字段	cluster.conf 属性	描述
名称	**name**	连接到集群的 APC 设备名称，在该设备中记录使用 SNMP 协议的 fence 守护进程日志。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址或者主机名。
UDP/TCP 端口	**udpport**	用来与该设备连接的 UDP/TCP 端口，默认值为 161。

luci 字段	cluster.conf 属性	描述
登录	**login**	访问该设备的登录名称。
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
SNMP 版本	**snmp_version**	要使用的 SNMP 版本（1，2c，3），默认值为 1。
SNMP 社区	**community**	SNMP 社区字符串。默认值为 **private**。
SNMP 安全等级	**snmp_sec_level**	SNMP 安全等级（noQuthNoPriv、authNoPriv、authPriv）。
SNMP 认证协议	**snmp_auth_prot**	SNMP 认证协议（MD5、SHA）。
SNMP 隐私协议	**snmp_priv_prot**	SNMP 隐私协议（DES、AES）
SNMP 隐私协议密码	**snmp_priv_passwd**	SNMP 隐私协议密码。
SNMP 隐私协议脚本	**snmp_priv_passwd_script**	该脚本为 SNMP 隐私协议提供密码。使用这个参数取代「**SNMP 隐私协议密码**」参数。
电源等待	**power_wait**	执行 power off 或者 power on 命令后要等待的秒数。
端口（出口）号	**port**	该端口

表 A.4 “Brocade 光纤开关” 列出 **fence_brocade** 使用的 fence 设备参数，Brocade FC 开关的 fence 代理。

表 A.4. Brocade 光纤开关

luci 字段	cluster.conf 属性	描述
名称	**name**	连接到该集群的 Brocade 设备名称。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址。

luci 字段	cluster.conf 属性	描述
登录	**login**	访问该设备的登录名称。
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
端口	**port**	开关插座数。

表 A.5 “Cisco MDS” 列出 **fence_cisco_mds** 使用的 fence 设备参数，Cisco MDS 的 fence 代理。

表 A.5. Cisco MDS

luci 字段	cluster.conf 属性	描述
名称	**name**	启用 SNMP 的 Cisco MDS 9000 系列设备的名称。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址或者主机名。
UDP/TCP 端口	**udpport**	用来与该设备连接的 UDP/TCP 端口，默认值为 161。
登录	**login**	访问该设备的登录名称。
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
端口（出口）号	**port**	该端口
SNMP 版本	**snmp_version**	要使用的 SNMP 版本（1、2c、3）。
SNMP 社区	**community**	SNMP 社区字符串。
SNMP 安全等级	**snmp_sec_level**	SNMP 安全等级（noQuthNoPriv、authNoPriv、authPriv）。
SNMP 认证协议	**snmp_auth_prot**	SNMP 认证协议（MD5、SHA）。

luci 字段	cluster.conf 属性	描述
SNMP 隐私协议	snmp_priv_prot	SNMP 隐私协议（DES、AES）
SNMP 隐私协议密码	snmp_priv_passwd	SNMP 隐私协议密码。
SNMP 隐私协议脚本	snmp_priv_passwd_script	该脚本为 SNMP 隐私协议提供密码。使用这个参数取代「SNMP 隐私协议密码」参数。
电源等待	power_wait	执行 power off 或者 power on 命令后要等待的秒数。

表 A.6 “Cisco UCS” 列出 **fence_cisco_ucs** 使用的 fence 设备参数，Cisco UCS 的 fence 代理。

表 A.6. Cisco UCS

luci 字段	cluster.conf 属性	描述
名称	name	Cisco UCS 设备名称。
IP 地址或者主机名	ipaddr	为该设备分配的 IP 地址或者主机名。
IP 端口（可选）	ipport	用来连接到该设备的 TCP 端口。
登录	login	访问该设备的登录名称。
密码	passwd	用来验证到该设备的连接的密码。
Password 脚本（可选）	passwd_script	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「Password」参数。
使用 SSL	ssl	使用 SSL 连接与该设备沟通。
子机构	suborg	访问子机构所需的附加路径。
端口（出口）号	port	虚拟机名称。
电源等待	power_wait	执行 power off 或者 power on 命令后要等待的秒数。

表 A.7 “Dell DRAC 5” 列出 **fence_drac5** 使用的 fence 设备参数，Dell DRAC 5 的 fence 代理。

表 A.7. Dell DRAC 5

luci 字段	cluster.conf 属性	描述
名称	**name**	分配给 DRAC 的名称。
IP 地址或者主机名	**ipaddr**	分配给 DRAC 的 IP 地址或者主机名。
IP 端口（可选）	**ipport**	用来连接到该设备的 TCP 端口。
登录	**login**	访问 DRAC 的登录名
密码	**passwd**	验证到 DRAC 的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代**「Password」**参数。
使用 SSH	**secure**	说明该系统将使用 SSH 访问该设备。
到 SSH 识别文件的路径	**identity_file**	SSH 的识别文件。
模块名称	**module_name**	（可选）当您有多个 DRAC 模块时用于这个 DRAC 的模块名。
强制命令提示	**cmd_prompt**	提示要使用的命令。默认值为 '\$'。
电源等待	**power_wait**	执行 power off 或者 power on 命令后要等待的秒数。

表 A.8 “Eaton 网络电源控制器（SNMP 接口）（红帽企业版 Linux 6.4 及之后的版本）” 中列出了 **fence_eaton_snmp** 使用的 fence 设备参数，该 fence 代理用于使用 SNMP 网络电源开关的 Eaton 设备。

表 A.8. Eaton 网络电源控制器（SNMP 接口）（红帽企业版 Linux 6.4 及之后的版本）

luci 字段	cluster.conf 属性	描述
名称	**name**	连接该集群的 Eaton 网络电源开关名称。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址或者主机名。
UDP/TCP 端口（可选）	**udpport**	用来与该设备连接的 UDP/TCP 端口，默认值为 161。

luci 字段	cluster.conf 属性	描述
登录	**login**	访问该设备的登录名称。
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
SNMP 版本	**snmp_version**	要使用的 SNMP 版本（1，2c，3），默认值为 1。
SNMP 社区	**community**	SNMP 社区字符串。默认值为 **private**。
SNMP 安全等级	**snmp_sec_level**	SNMP 安全等级（noQuthNoPriv、authNoPriv、authPriv）。
SNMP 认证协议	**snmp_auth_prot**	SNMP 认证协议（MD5、SHA）。
SNMP 隐私协议	**snmp_priv_prot**	SNMP 隐私协议（DES、AES）
SNMP 隐私协议密码	**snmp_priv_passwd**	SNMP 隐私协议密码。
SNMP 隐私协议脚本	**snmp_priv_passwd_script**	该脚本为 SNMP 隐私协议提供密码。使用这个参数取代「**SNMP 隐私协议密码**」参数。
电源等待（秒）	**power_wait**	执行 power off 或者 power on 命令后要等待的秒数。
端口（出口）号	**port**	物理插件数或者虚拟机名称。总是需要这个参数。

表 A.9 “Egenera SAN 控制器” 列出 **fence_egenera** 使用的 fence 设备参数，Egenera BladeFrame 的 fence 代理。

表 A.9. Egenera SAN 控制器

luci 字段	cluster.conf 属性	描述
名称	**name**	连接到该集群的 Egenera BladeFrame 设备的名称。

luci 字段	cluster.conf 属性	描述
CServer	**cserver**	分配给该设备的主机名（以及可选用户名，格式为 **username@hostname**）。详情请参考 fence_egenera(8) man page。
ESH 路径（可选）	**esh**	到 cserver 中到 esh 命令的路径（默认为 /opt/panmgr/bin/esh）
用户名	**user**	登录名。默认值为 **root**。
lpan	**lpan**	该设备的局域网逻辑进程（LPAN）。
pserver	**pserver**	该设备的处理刀片服务器（pserver）名称。

表 A.10 “ePowerSwitch” 列出 **fence_eps** 使用的 fence 设备参数，ePowerSwitch 的 fence 代理。

表 A.10. ePowerSwitch

luci 字段	cluster.conf 属性	描述
名称	**name**	连接到该集群的 ePowerSwitch 设备名称。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址或者主机名。
登录	**login**	访问该设备的登录名称。
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
隐藏页名称	**hidden_page**	为该设备隐藏的页面名称。
端口（出口）号	**port**	物理插件数或者虚拟机名称。

表 A.11 “Fence virt” 列出 **fence_virt** 使用的 fence 设备参数，Fence virt fence 设备的 fence 代理。

表 A.11. Fence virt

luci 字段	cluster.conf 属性	描述
名称	**name**	Fence virt fence 设备的名称。
串口设备	**serial_device**	在主机中，串口设备必须与每个域的配置文件映射。有关详情请参考 **fence_virt.conf** man page。如果指定了该字段，则 **fence_virt** fencing 代理会使用串口模式操作。不指定该值，则 **fence_virt** fencing 代理会使用 VM 通道模式操作。
串口参数	**serial_params**	串口参数。默认为 115200，8N1。
虚拟机频道 IP 地址	**channel_address**	频道 IP。默认值为 10.0.2.179。
端口或域（已弃用）	**port**	要 fence 的虚拟机（域 UUID 或者名称）。
	ipport	频道端口。默认值为 1229，在使用 **luci** 配置这个 fence 设备时使用这个值。

表 A.12 “富士通-西门子远程查看服务栏（RSB）” 列出 **fence_rsb** 使用的 fence 设备参数，富士通-西门子 RSB 的 fence 代理。

表 A.12. 富士通-西门子远程查看服务栏（RSB）

luci 字段	cluster.conf 属性	描述
名称	**name**	作为 fence 设备使用的 RSB 名称。
IP 地址或者主机名	**ipaddr**	分配给该设备的主机名。
登录	**login**	访问该设备的登录名称。
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
TCP 端口	**ipport**	Telnet 服务侦听的端口号。默认值为 3172。

表 A.13 “HP 刀片机系统（红帽企业版 Linux 6.4 及之后的版本）” 列出 **fence_hpblade** 使用的 fence 设备参数，HP 刀片机系统的 fence 代理。

表 A.13. HP 刀片机系统（红帽企业版 Linux 6.4 及之后的版本）

luci 字段	cluster.conf 属性	描述
名称	**name**	分配给连接到该集群的 HP 刀片机系统设备的名称。
IP 地址或者主机名	**ipaddr**	分配给该 HP 刀片机系统设备的 IP 地址或者主机名。
IP 端口（可选）	**ipport**	用来连接到该设备的 TCP 端口。
登录	**login**	用来访问该 HP 刀片机系统设备的登录名。这个参数是必须的。
密码	**passwd**	用来验证到该 fence 设备连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
强制命令提示	**cmd_prompt**	提示要使用的命令。默认值为 '\$'。
缺少端口将返回 OFF 而不是失败	**missing_as_off**	缺少端口将返回 OFF 而不是失败。
电源等待（秒）	**power_wait**	执行 power off 或者 power on 命令后要等待的秒数。
使用 SSH	**secure**	说明该系统将使用 SSH 访问该设备。
到 SSH 识别文件的路径	**identity_file**	SSH 的识别文件。

表 A.14 “HP iLO/iLO2 (Integrated Lights Out)” 列出 **fence_ilo** 使用的 fence 设备参数，HP iLO 设备的 fence 代理。

表 A.14. HP iLO/iLO2 (Integrated Lights Out)

luci 字段	cluster.conf 属性	描述
名称	**name**	支持惠普 iLO 的服务器名称。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址或者主机名。
IP 端口（可选）	**ipport**	用来连接该设备的 TCP 端口。

luci 字段	cluster.conf 属性	描述
登录	**login**	访问该设备的登录名称。
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
电源等待	**power_wait**	执行 power off 或者 power on 命令后要等待的秒数。

表 A.15 “惠普 iLO（Integrated Lights Out）MP” 列出 **fence_ilo_mp** 使用的 fence 设备参数，HP iLO MP 设备的 fence 代理。

表 A.15. 惠普 iLO（Integrated Lights Out）MP

luci 字段	cluster.conf 属性	描述
名称	**name**	支持惠普 iLO 的服务器名称。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址或者主机名。
IP 端口（可选）	**ipport**	用来连接该设备的 TCP 端口。
登录	**login**	访问该设备的登录名称。
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
使用 SSH	**secure**	说明该系统将使用 SSH 访问该设备。
到 SSH 识别文件的路径	**identity_file**	SSH 的识别文件。
强制命令提示	**cmd_prompt**	提示要使用的命令。默认值为 ’MP>’, ’hpiLO->’。
电源等待	**power_wait**	执行 power off 或者 power on 命令后要等待的秒数。

表 A.16 “IBM 刀片服务器” 列出 **fence_bladecenter** 使用的 fence 设备参数，IBM BladeCenter 的 fence 代理。

表 A.16. IBM 刀片服务器

luci 字段	cluster.conf 属性	描述
名称	**name**	连接该集群的 IBM 刀片服务器设备名称。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址或者主机名。
IP 端口（可选）	**ipport**	用来连接该设备的 TCP 端口。
登录	**login**	访问该设备的登录名称。
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
电源等待	**power_wait**	执行 power off 或者 power on 命令后要等待的秒数。
使用 SSH	**secure**	表示系统将使用 SSH 访问该设备。
到 SSH 识别文件的路径	**identity_file**	SSH 的识别文件。

表 A.17 “IBM 刀片服务器 SNMP” 列出 **fence_ibmblade** 使用的 fence 设备参数，SNMP 的 fence 代理。

表 A.17. IBM 刀片服务器 SNMP

luci 字段	cluster.conf 属性	描述
名称	**name**	连接该集群的 IBM 刀片服务器 SNMP 名称。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址或者主机名。
UDP/TCP 端口（可选）	**udpport**	用来连接该设备的 UDP/TCP 端口，默认值为 161。
登录	**login**	访问该设备的登录名称。
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。

luci 字段	cluster.conf 属性	描述
SNMP 版本	snmp_version	要使用的 SNMP 版本（1，2c，3），默认值为 1。
SNMP 社区	community	SNMP 社区字符串。
SNMP 安全等级	snmp_sec_level	SNMP 安全等级（noQuthNoPriv、authNoPriv、authPriv）。
SNMP 认证协议	snmp_auth_prot	SNMP 认证协议（MD5、SHA）。
SNMP 隐私协议	snmp_priv_prot	SNMP 隐私协议（DES、AES）
SNMP 隐私协议密码	snmp_priv_passwd	SNMP 隐私协议密码。
SNMP 隐私协议脚本	snmp_priv_passwd_script	该脚本为 SNMP 隐私协议提供密码。使用这个参数取代「**SNMP 隐私协议密码**」参数。
电源等待	power_wait	执行 power off 或者 power on 命令后要等待的秒数。
端口	port	物理插件数或者虚拟机名称。

表 A.18 “IBM iPDU（红帽企业版 Linux 6.4 及之后的版本）” 列出 **fence_ipdu** 使用的 fence 设备参数，该 fence 代理用于 SNMP 设备的 iPDU。

表 A.18. IBM iPDU（红帽企业版 Linux 6.4 及之后的版本）

luci 字段	cluster.conf 属性	描述
名称	name	连接到集群的 IBM iPDU 设备名称，在该设备中记录使用 SNMP 协议的 fence 守护进程日志。
IP 地址或者主机名	ipaddr	为该设备分配的 IP 地址或者主机名。
UDP/TCP 端口	udpport	用来与该设备连接的 UDP/TCP 端口，默认值为 161。
登录	login	访问该设备的登录名称。
密码	passwd	用来验证到该设备的连接的密码。

luci 字段	cluster.conf 属性	描述
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
SNMP 版本	**snmp_version**	要使用的 SNMP 版本（1，2c，3），默认值为 1。
SNMP 社区	**community**	SNMP 社区字符串。默认值为 **private**。
SNMP 安全等级	**snmp_sec_level**	SNMP 安全等级（noQuthNoPriv、authNoPriv、authPriv）。
SNMP 认证协议	**snmp_auth_prot**	SNMP 认证协议（MD5、SHA）。
SNMP 隐私协议	**snmp_priv_prot**	SNMP 隐私协议（DES、AES）
SNMP 隐私协议密码	**snmp_priv_passwd**	SNMP 隐私协议密码。
SNMP 隐私协议脚本	**snmp_priv_passwd_script**	该脚本为 SNMP 隐私协议提供密码。使用这个参数取代「**SNMP 隐私协议密码**」参数。
电源等待	**power_wait**	执行 power off 或者 power on 命令后要等待的秒数。
端口	**port**	该端口

表 A.19 “IF MIB” 列出 **fence_ifmib** 使用的 fence 设备参数，IF-MIB 设备的 fence 代理。

表 A.19. IF MIB

luci 字段	cluster.conf 属性	描述
名称	**name**	连接该集群的 IF MIB 设备名称。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址或者主机名。
UDP/TCP 端口（可选）	**udpport**	用来与该设备连接的 UDP/TCP 端口，默认值为 161。
登录	**login**	访问该设备的登录名称。

luci 字段	cluster.conf 属性	描述
密码	passwd	用来验证到该设备的连接的密码。
Password 脚本（可选）	passwd_script	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「Password」参数。
SNMP 版本	snmp_version	要使用的 SNMP 版本（1，2c，3），默认值为 1。
SNMP 社区	community	SNMP 社区字符串。
SNMP 安全等级	snmp_sec_level	SNMP 安全等级（noQuthNoPriv、authNoPriv、authPriv）。
SNMP 认证协议	snmp_auth_prot	SNMP 认证协议（MD5、SHA）。
SNMP 隐私协议	snmp_priv_prot	SNMP 隐私协议（DES、AES）
SNMP 隐私协议密码	snmp_priv_passwd	SNMP 隐私协议密码。
SNMP 隐私协议脚本	snmp_priv_passwd_script	该脚本为 SNMP 隐私协议提供密码。使用这个参数取代「SNMP 隐私协议密码」参数。
电源等待	power_wait	执行 power off 或者 power on 命令后要等待的秒数。
端口	port	物理插件数或者虚拟机名称。

表 A.20 “Intel 模块化” 列出 **fence_intelmodular** 使用的 fence 设备参数，Intel Modular 的 fence 代理。

表 A.20. Intel 模块化

luci 字段	cluster.conf 属性	描述
名称	name	连接到该集群的 Intel 模块设备名称。
IP 地址或者主机名	ipaddr	为该设备分配的 IP 地址或者主机名。
登录	login	访问该设备的登录名称。

luci 字段	cluster.conf 属性	描述
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
SNMP 版本	**snmp_version**	要使用的 SNMP 版本（1，2c，3），默认值为 1。
SNMP 社区	**community**	SNMP 社区字符串。默认值为 **private**。
SNMP 安全等级	**snmp_sec_level**	SNMP 安全等级（noQuthNoPriv、authNoPriv、authPriv）。
SNMP 认证协议	**snmp_auth_prot**	SNMP 认证协议（MD5、SHA）。
SNMP 隐私协议	**snmp_priv_prot**	SNMP 隐私协议（DES、AES）
SNMP 隐私协议密码	**snmp_priv_passwd**	SNMP 隐私协议密码。
SNMP 隐私协议脚本	**snmp_priv_passwd_script**	该脚本为 SNMP 隐私协议提供密码。使用这个参数取代「**SNMP 隐私协议密码**」参数。
电源等待	**power_wait**	执行 power off 或者 power on 命令后要等待的秒数。
端口	**port**	物理插件数或者虚拟机名称。

表 A.21 “IPMI（智能平台管理界面）LAN” 列出 **fence_ipmilan** 使用的 fence 设备参数，IPMI 通过 LAN 的 fence 代理。

表 A.21. IPMI（智能平台管理界面）LAN

luci 字段	cluster.conf 属性	描述
名称	**name**	连接到该集群的 IPMI LAN 设备名称。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址或者主机名。
登录	**login**	可为给定 IPMI 端口发出 power on/off 命令的用户登录名。

luci 字段	cluster.conf 属性	描述
密码	**passwd**	用来验证到 IPMI 端口连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
认证类型	**auth**	IPMI LAN 认证类型：**none**, **password** 或者 **md5**。
使用 Lanplus	**lanplus**	**True** 或者 **1**。如果空白，则该值为 **False**。
要使用的加密套接字	**cipher**	用于 IPMIv2 lanplus 连接的远程服务器验证、完整性以及加密算法。
特权等级	**privlvl**	IPMI 设备中的特权等级。

表 A.22 “RHEV-M REST API（RHEL 6.2 及之后的版本，RHEV 3.0 及之后的版本）” 列出 **fence_rhevm** 使用的 fence 设备参数，RHEV-M REST API 的 fence 代理。

表 A.22. RHEV-M REST API（RHEL 6.2 及之后的版本，RHEV 3.0 及之后的版本）

luci 字段	cluster.conf 属性	描述
名称	**name**	RHEV-M REST API fencing 设备名称。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址或者主机名。
IP 端口（可选）	**ipport**	用来连接该设备的 TCP 端口。
登录	**login**	访问该设备的登录名称。
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
使用 SSL	**ssl**	使用 SSL 连接与该设备沟通。
电源等待	**power_wait**	执行 power off 或者 power on 命令后要等待的秒数。
端口	**port**	物理插件数或者虚拟机名称。

表 A.23 “SCSI Fencing” 列出 **fence_scsi** 使用的 fence 设备参数，为 SCSI 持续保留的 fnce 代理。

注意

支持使用 SCSI 永久保留作为 fence 方法但有以下限制：

- 当使用 SCSI fencing 时，集群中的所有节点必须使用同一设备注册，这样每个节点都可删除另一个节点的注册密钥，即该节点在所有设备中用来注册的密钥。
- 用于集群卷的设备应该是完整的 LUN，不是分区。SCSI 永久保留适用于整个 LUN，即控制到每个 LUN 的访问，而不是对独立分区的访问。

表 A.23. SCSI Fencing

luci 字段	cluster.conf 属性	描述
名称	**name**	SCSI fence 设备名称。
Node name		
当前动作的按键		（覆盖节点名称）

表 A.24 “VMware Fencing（SOAP 接口）（红帽企业版 Linux 6.2 及之后的版本）” 列出 **fence_vmware_soap** 使用的 fence 设备参数，VMWare 通过 SOAP API 的 fence 代理。

表 A.24. VMware Fencing（SOAP 接口）（红帽企业版 Linux 6.2 及之后的版本）

luci 字段	cluster.conf 属性	描述
名称	**name**	与 fencing 设备映射的虚拟机名称。
IP 地址或者主机名	**ipaddr**	为该设备分配的 IP 地址或者主机名。
IP 端口（可选）	**ipport**	用来连接该设备的 TCP 端口。
登录	**login**	访问该设备的登录名称。
密码	**passwd**	用来验证到该设备的连接的密码。
Password 脚本（可选）	**passwd_script**	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「**Password**」参数。
分隔符	**separator**	操作列表创建的 CSV 的分隔符。默认只为逗号（,）。
电源等待	**power_wait**	执行 power off 或者 power on 命令后要等待的秒数。

luci 字段	cluster.conf 属性	描述
虚拟机名称	port	以清单路径格式显示的虚拟机名称（例如：/datacenter/vm/Discovered_virtual_machine/myMachine）。
虚拟机 UUID	uuid	要 fence 的虚拟机 UUID。
使用 SSL	ssl	使用 SSL 连接与该设备沟通。

表 A.25 “WTI 电源开关” 列出 fence_wti 使用的 fence 设备参数，WTI 网络电源开关的 fence 代理。

表 A.25. WTI 电源开关

luci 字段	cluster.conf 属性	描述
名称	name	连接到集群的 WTI 电源开关的名称。
IP 地址或者主机名	ipaddr	分配给该设备的 IP 或者主机名地址。
IP 端口（可选）	ipport	用来连接到该设备的 TCP 端口。
登录	login	访问该设备的登录名称。
密码	passwd	用来验证到该设备的连接的密码。
Password 脚本（可选）	passwd_script	为访问该 fence 设备提供密码的脚本。使用这个参数可取代「Password」参数。
端口	port	物理插件数或者虚拟机名称。
强制命令提示	cmd_prompt	使用的命令提示。默认值为 ['RSM>', '>MPC', 'IPS>', 'TPS>', 'NBB>', 'NPS>', 'VMR>']。
电源等待	power_wait	执行 power off 或者 power on 命令后要等待的秒数。
使用 SSH	secure	表示系统将使用 SSH 访问该设备。
到 SSH 识别文件的路径	identity_file	SSH 的识别文件。

附录 B. HA 资源参数

本附录提供 HA 资源参数的描述。您可以使用 **luci** 配置这些参数，方法是使用 **ccs** 命令，或者编辑 **/etc/cluster/cluster.conf** 文件。表 B.1 “HA 资源小结” 列出了这些资源、其对应的资源代理以及其他包含参数描述表格的参考。有关资源代理详情请查看任意集群节点中的 **/usr/share/cluster** 文件。

除在这个附录中描述的资源代理外，**/usr/share/cluster** 目录还包括资源组的仿制 OCF 脚本 **service.sh**。有关包含在这个脚本中的参数详情请参考 **service.sh** 脚本。

有关 **cluster.conf** 元素和属性的完整列表及描述，请参考 **/usr/share/cluster/cluster.rng** 中的集群方案，以及 **/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html** 中的注释方案（例如：**/usr/share/doc/cman-3.0.12/cluster_conf.html**）。

表 B.1. HA 资源小结

资源	资源代理	参数描述参考
Apache	apache.sh	表 B.2 “Apache 服务器”
Condor 事务	condor.sh	表 B.3 “Condor 事务”
文件系统	fs.sh	表 B.4 “文件系统”
GFS2 文件系统	clusterfs.sh	表 B.5 “GFS2”
IP 地址	ip.sh	表 B.6 “IP 地址”
HA LVM	lvm.sh	表 B.7 “HA LVM”
MySQL	mysql.sh	表 B.8 “MySQL”
NFS 客户端	nfsclient.sh	表 B.9 “NFS 客户端”
NFS 导出	nfsexport.sh	表 B.10 “NFS 导出”
NFS 服务器	nfsserver.sh	表 B.11 “NFS 服务器”
NFS/CIFS 挂载	netfs.sh	表 B.12 “NFS/CIFS 挂载”
Open LDAP	openldap.sh	表 B.13 “Open LDAP”
Oracle 10g/11g 故障切换事务	oracledb.sh	表 B.14 “Oracle 10g/11G 故障切换事务”
Oracle 10g 故障切换事务	orainstance.sh	表 B.15 “Oracle 10g 故障切换事务”
Oracle 10g 侦听程序	oralistener.sh	表 B.16 “Oracle 10g 侦听程序”

资源	资源代理	参数描述参考
PostgreSQL 8	postgres-8.sh	表 B.17 “PostgreSQL 8”
SAP 数据库	SAPDatabase	表 B.18 “SAP 数据库”
SAP 事务	SAPInstance	表 B.19 “SAP 事务”
Samba	samba.sh	表 B.20 “Samba 服务器”
脚本	script.sh	表 B.21 “脚本”
Sybase ASE	ASEHAagent.sh	表 B.22 “Sybase ASE 故障切换事务”
Tomcat 6	tomcat-6.sh	表 B.23 “Tomcat 6”
虚拟机	vm.sh	表 B.24 “虚拟机” 注：如果主机集群可支持虚拟机，则 **Luci** 会将其显示为虚拟服务。

表 B.2. Apache 服务器

luci 字段	`cluster.conf` 属性	描述
名称	`name`	Apache 服务名称
服务器 Root	`server_root`	默认值为 `/etc/httpd`。
配置文件	`config_file`	指定 Apache 配置文件。默认值为 `/etc/httpd/conf`。
httpd 选项	`httpd_options`	`httpd` 的其他命令行选项。
关闭等待（秒为单位）	`shutdown_wait`	指定等待正确关闭服务的秒数。

表 B.3. Condor 事务

字段	luci 字段	`cluster.conf` 属性
事务名称	`name`	指定 Condor 事务唯一名称。
Confor 子系统类型	`type`	为这个事务指定 Condor 子系统类型：`schedd`、`job_server` 或者 `query_server`。

表 B.4. 文件系统

luci 字段	cluster.conf 属性	描述
名称	**name**	指定文件系统资源名称。
文件系统类型	**fstype**	如果没有指定，**mount** 会尝试确定文件系统类型。
挂载点	**mountpoint**	挂载这个文件系统的文件系统结构路径。
设备、文件系统标签或者 UUID	**device**	指定与这个文件系统资源关联的设备。可以是块设备、文件系统标签或者文件系统的 UUID。
挂载选项	**options**	挂载选项，即在挂载文件系统时使用的选项。这要根据具体文件系统决定。支持的挂载选项请查看 ***mount****(8)* man page。
文件系统 ID（自选）	**fsid**	 **注意** ***File System ID*** 只可由 NFS 服务使用 当创建新的文件系统资源时，您可以让此字段保持空白。保持空白后会在配置过程中提交参数后自动分配 file system ID。如果您要具体指定 file system ID，请在此字段中指定。
强制卸载	**force_unmount**	如果启用该功能，则会强制卸载文件系统。默认设置为 ***disabled***。***Force Unmount*** 会杀死使用该挂载点的所有进程以便在其尝试卸载时释放挂载点。
强制 fsck	**force_fsck**	如果启用该选项，则会在挂载前在文件系统中运行 **fsck**。默认设置为 ***disabled***。
启用 NFS 守护进程以及 lockd 临时规避方法（红帽企业版 Linux 6.4 以及之后的版本）	**nfsrestart**	如果使用 NFS 导出您的文件系统，并偶尔无法卸载（在关机或者服务重新定位的过程中），设定这个选项将在执行卸载操作前取消所有文件系统参考。设定这个选项需要您启用 **强制卸载** 选项，并一定不能与 **NFS 服务器**资源一同使用。您应只将这个选项设定为最后的手段，因为这是卸载文件系统的强制手段。
使用快速状态查看	**quick_status**	启用后，请执行快速状态查看。
卸载失败时请重启主机节点	**self_fence**	如果已启用，则在卸载这个文件系统失败时重启该节点。**filesystem** 资源代理使用值 1、**yes**、**on** 或者 **true** 启用这个参数；使用 0、**no**、**off** 或者 **false** 禁用这个参数。默认设置为 ***disabled***。

表 B.5. GFS2

luci 字段	cluster.conf 属性	描述
名称	**name**	文件系统资源名称
挂载点	**mountpoint**	挂载文件系统资源的路径。
设备、文件系统标签或者 UUID	**device**	与文件系统资源关联的设备文件。
文件系统类型	**fstype**	在 **luci** 中设定 GFS2
挂载选项	**options**	挂载选项
文件系统 ID（自选）	**fsid**	 **注意** ***File System ID*** 只可由 NFS 服务使用 当创建新的 GFS2 资源时，您可以让此字段保持空白。保持空白后会在配置过程中提交参数后自动分配 file system ID。如果您要具体指定 file system ID，请在此字段中指定。
强制卸载	**force_unmount**	如果启用该选项，则会强制卸载文件系统。默认设置为 ***disabled***。***Force Unmount*** 会杀死使用该挂载点的所有进程以便在其尝试卸载时释放挂载点。使用 GFS2 资源时，服务停止时 *不卸载* 挂载点除非 ***Force Unmount*** 是 *enabled*。
启用 NFS 守护进程以及 lockd 临时规避方法（红帽企业版 Linux 6.4 以及之后的版本）	**nfsrestart**	如果使用 NFS 导出您的文件系统，并偶尔无法卸载（在关机或者服务重新定位的过程中），设定这个选项将在执行卸载操作前取消所有文件系统参考。设定这个选项需要您启用 **强制卸载** 选项，并一定不能与 **NFS 服务器**资源一同使用。您应只将这个选项设定为最后的手段，因为这是卸载文件系统的强制手段。
卸载失败时请重启主机节点	**self_fence**	如果已启用，且在卸载这个文件系统时失败，则该节点将立即重启。一般它与 force-unmount一同使用，但不是必须的。**filesystem** 资源代理使用值 1、**yes**、**on** 或者 **true** 启用这个参数；使用 0、**no**、**off** 或者 **false** 禁用这个参数。

表 B.6. IP 地址

luci 字段	cluster.conf 属性	描述
IP 地址，子网掩码字节	**address**	该资源的 IP 地址（和自选子网页面字节）。根据 CIDR 表示法，子网页面字节或者网络前缀长度可紧跟该地址，并使用斜线作为分隔符（例如：10.1.1.1/8）。这是一个虚拟 IP 地址。支持 IPv4 和 IPv6 地址，因为 NIC 链接监控每个 IP 地址。

luci 字段	cluster.conf 属性	描述
监控链接	**monitor_link**	如果没有这个 IP 地址绑定的 NIC 的链接，启用此选项将导致状态检查失败。
禁用静态路由更新	**disable_rdisc**	禁止使用 RDISC 协议更新路由。
删除 IP 地址多少秒后进入睡眠状态	**sleeptime**	指定睡眠状态时间（单位为秒）。

表 B.7. HA LVM

luci 字段	cluster.conf 属性	描述
名称	**name**	这个 LVM 资源的唯一名称。
卷组名称	**vg_name**	被管理卷组的说明性名称。
逻辑卷名称（可选）	**lv_name**	被管理的逻辑卷名称。如果该卷组中被管理的逻辑卷在一个以上，则这个参数是可选的。
如无法清除 LVM 标签则 fence 该节点	**self_fence**	如果无法清除 LVM 标签则需要 fence 该节点。LVM 资源代理使用 1 或者 **yes** 启用此参数，使用 0 或者 **no** 禁用它。

表 B.8. MySQL

luci 字段	cluster.conf 属性	描述
名称	**name**	指定 MySQL 服务器资源名称。
配置文件	**config_file**	指定配置文件。默认值为 **/etc/my.cnf**。
侦听地址	**listen_address**	指定 MySQL 服务器的 IP 地址。如果没有提供 IP 地址，则会为该设备使用第一个 IP 地址。
mysqld 选项	**mysqld_options**	**httpd** 的其他命令行选项。
启动等待（秒为单位）	**startup_wait**	指定等待正确终止服务启动的秒数。

luci 字段	cluster.conf 属性	描述
关闭等待（秒为单位）	**shutdown_wait**	指定等待正确关闭服务的秒数。

表 B.9. NFS 客户端

luci 字段	cluster.conf 属性	描述
名称	**name**	这是客户端用来在资源树中进行参考的符号名。这与 ***Target*** 选项*不一样*。
目标主机名、通配符或网络组群	**target**	这是您要执行挂载的服务器。可使用主机名、通配符（基于 IP 地址或者主机名）或者定义主机，或者导出主机的网络组群定义该服务器。
允许恢复这个 NFS 客户端	**allow_recover**	允许恢复。
选项	**options**	为这个客户端定义一组选项 — 例如：额外客户端访问权力。有关详情请参考 ***exports*** *(5)* man page，*《常规选项》*。

表 B.10. NFS 导出

luci 字段	cluster.conf 属性	描述
名称	**name**	资源的说明性名称。NFS 导出资源确定 NFS 守护进程正在运行。它可重复使用，通常只需要 NFS 导出资源。  **注意** 为 NFS 导出命名以便区别于其他 NFS 资源。

表 B.11. NFS 服务器

luci 字段	cluster.conf 属性	描述
名称	**name**	NFS 服务器资源的描述性名称。NFS 服务器资源对将 NFSv4 文件系统导出到客户端非常有帮助。因为 NFSv4 方法起作用，所以每次在一个服务器中只能有一个 NFSv4 资源。另外，不可能在每个集群节点中同时还使用本地 NFS 事务时使用该 NFS 服务器资源。

表 B.12. NFS/CIFS 挂载

luci 字段	cluster.conf 属性	描述
名称	**name**	NFS 或者 CIFS 挂载的符号名。  **注意** 只有将集群服务配置为 NFS 客户端时才需要这个资源。
挂载点	**mountpoint**	挂载文件系统资源的路径。
主机	**host**	NFS/CIFS 服务器 IP 地址或者主机名。
NFS 导出目录名或者 CIFS 共享	**export**	NFS 导出目录名或者 CIFS 共享名称。
文件系统类型	**fstype**	文件系统类型： • ***NFS3*** — 指定使用默认 NFS 版本。这是默认设置。 • ***NFS v4*** — 指定使用 NFSv4 协议。 • ***CIFS*** — 指定使用 CIFS 协议。
强制卸载	**force_unmount**	如果启用 ***Force Unmount***，则集群会在服务停止时杀死所有使用该文件系统的进程。杀死所有使用该文件系统的进程可释放文件系统。另外，如果卸载失败则会重启该服务。
停止重新定位操作的过程中不卸载该文件系统。	**no_unmount**	如果启用，可指定在停止或者重新定位操作中不能卸载的文件系统。
选项	**options**	挂载选项。指定挂载选项列表。如果没有指定，则会使用 **-o sync** 挂载文件系统。

表 B.13. Open LDAP

luci 字段	cluster.conf 属性	描述
名称	**name**	指定文件系统名称用于日志记录或者其它目的。
配置文件	**config_file**	指定配置文件的绝对路径。默认值为 **/etc/openldap/slapd.conf**。
URL 列表	**url_list**	默认值为 **ldap:///**。

luci 字段	cluster.conf 属性	描述
slapd Options	**slapd_options**	**slapd** 的其它命令行选项。
关闭等待（秒为单位）	**shutdown_wait**	指定等待正确关闭服务的秒数。

表 B.14. Oracle 10g/11G 故障切换事务

luci 字段	cluster.conf 属性	描述
Oracle 事务的事务名称（SID）	**name**	事务名称。
Oracle 用户名	**user**	这是在 Oracle AS 事务中运行的 Oracle 用户的用户名。
Oracle 应用程序主目录	**home**	这是 Oracle（应用程序，不是用户）主目录。您安装 Oracle 后就会配置这个目录。
Oracle 安装类型	**type**	Oracle 安装类型。默认：**10g**，只有数据库事务以侦听程序 **base**，数据库、侦听程序、企业版管理器以及 ISQL*PLUS：**base-em**（或者 **10g**），或者互联网应用程序服务器（基础设施）：**ias**（或者 **10g-ias**）。
虚拟主机名（可选）	**vhost**	与 Oracle 10g 安装主机名匹配的虚拟主机名。注：在启动/停止某个 oracledb 资源的过程中，您的主机名可能会临时更改为这个主机名。因此，您应该只将 oracledb 资源配置为专有服务的一部分。

表 B.15. Oracle 10g 故障切换事务

luci 字段	cluster.conf 属性	描述
Oracle 事务的事务名称（SID）	**name**	事务名称。
Oracle 用户名	**user**	这是在作为 Oracle 事务运行的 Oracle 用户的用户名。
Oracle 应用程序主目录	**home**	这是 Oracle（应用程序，不是用户）主目录。您安装 Oracle 后就会配置这个目录。

luci 字段	cluster.conf 属性	描述
Oracle 侦听程序列表（可选，使用空格分开）	**listeners**	可使用数据库事务启动的 Oracle 侦听程序列表。侦听程序名称使用空格分开。默认为空白，即禁用侦听程序。
锁定文件路径（可选）	**lockfile**	用来检查 Oracle 是否应该运行的锁定文件位置。默认为 **/tmp** 目录中的某个位置。

表 B.16. Oracle 10g 侦听程序

luci 字段	cluster.conf 属性	描述
侦听程序名称	**name**	侦听程序名称。
Oracle 用户名	**user**	这是在作为 Oracle 事务运行的 Oracle 用户的用户名。
Oracle 应用程序主目录	**home**	这是 Oracle（应用程序，不是用户）主目录。您安装 Oracle 后就会配置这个目录。

表 B.17. PostgreSQL 8

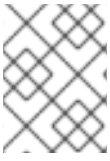
luci 字段	cluster.conf 属性	描述
名称	**name**	指定文件系统名称用于日志记录或者其它目的。
配置文件	**config_file**	定义配置文件的绝对路径。默认值为 **/var/lib/pgsql/data/postgresql.conf**。
Postmaster 用户	**postmaster_user**	在 root 无法运行该数据库服务器时运行该服务器的用户。默认值为 postgres。
Postmaster 选项	**postmaster_options**	Postmaser 的其他命令行选项。
关闭等待（秒为单位）	**shutdown_wait**	指定等待正确关闭服务的秒数。

表 B.18. SAP 数据库

luci 字段	cluster.conf 属性	描述
SAP 数据库名称	**SID**	指定唯一 SAP 系统识别符。例如：P01。
SAP 可执行目录	**DIR_EXECUTABLE**	为 **sapstartsrv** 和 **sapcontrol** 指定完全限定路径。
数据库类型	**DBTYPE**	指定以下数据库类型之一：Oracle、DB6 或者 ADA。
Oracle 侦听程序名称	**NETSERVICENAME**	指定 Oracle TNS 侦听程序名称。
ABAP 栈尚未安装，只安装了 Java 栈。	**DBJ2EE_ONLY**	如果您没有在 SAP 数据库中安装 ABAP 栈，则启用这个参数。
应用程序等级监控	**STRICT_MONITORING**	激活应用程序等级监控
自动启动恢复	**AUTOMATIC_RECOVER**	启用或禁用自动启动恢复。
Java SDK 路径	**JAVE_HOME**	Java SDK 路径。
JDBC 驱动程序文件名	**DB_JARS**	JDBC 驱动程序文件名。
预启动脚本路径	**PRE_START_USEREXIT**	预启动脚本路径。
后启动脚本路径	**POST_START_USEREXIT**	后启动脚本路径。
预停止脚本路径	**PRE_STOP_USEREXIT**	预停止脚本路径
后停止脚本路径	**POST_STOP_USEREXIT**	后停止脚本路径
J2EE 事务启动目录	**DIR_BOOTSTRAP**	J2EE 事务引导程序目录的完全限定路径。例如：**/usr/sap/P01/J00/j2ee/cluster/bootstrap**。
J2EE 安全存储路径	**DIR_SECSTORE**	J2EE 安全存储目录的完全限定路径。例如：**/usr/sap/P01/SYS/global/security/lib/tools**。

表 B.19. SAP 事务

luci 字段	cluster.conf 属性	描述
SAP 事务名称	**InstanceName**	完全限定的 SAP 事务名称。例如：P01_DVEBMGS00_sapp01ci。
SAP 可执行目录	**DIR_EXECUTABLE**	**sapstartsrv** 和 **sapcontrol** 的完全限定路径。
包含 SAP 启动侧写的目录	**DIR_PROFILE**	SAP 启动侧写的完全限定路径。
SAP 启动侧写名称	**START_PROFILE**	SAP 启动侧写的指定名称。
指定检查启动状态前等待的秒数	**START_WAITTIME**	指定检查启动状态前等待的秒数（不等待 J2EE-Addin）。
启用自动启动恢复	**AUTOMATIC_RECOVER**	启用或禁用自动启动恢复。
预启动脚本路径	**PRE_START_USEREXIT**	预启动脚本路径。
后启动脚本路径	**POST_START_USEREXIT**	后启动脚本路径。
预停止脚本路径	**PRE_STOP_USEREXIT**	预停止脚本路径
后停止脚本路径	**POST_STOP_USEREXIT**	后停止脚本路径



注意

根据 表 B.20 “Samba 服务器”，当创建或者编辑集群服务时，直接将 Samba 服务资源而 *不是* 服务中的资源连接到该服务。

表 B.20. Samba 服务器

luci 字段	cluster.conf 属性	描述
名称	**name**	指定 Samba 服务器的名称。
配置文件	**config_file**	Samba 配置文件路径

luci 字段	cluster.conf 属性	描述
smbd 的其他命令行选项	**smbd_options**	smbd 的其他命令行选项。
nmbd 的其他命令行选项	**nmbd_options**	nmbd 的其他命令行选项。
关闭等待（秒为单位）	**shutdown_wait**	指定等待正确终止服务关闭的秒数。

表 B.21. 脚本

luci 字段	cluster.conf 属性	描述
名称	**name**	指定自定义用户脚本的名称。该脚本资源允许使用符合 LSB 标准的 init 脚本启动集群的服务。
脚本文件的完整路径	**file**	输入自定义脚本所在路径（例如：**/etc/init.d/*userscript***）。

表 B.22. Sybase ASE 故障切换事务

luci 字段	cluster.conf 属性	描述
事务名称	**name**	指定 Sybase ASE 资源的事务名称。
ASE 服务器名称	**server_name**	为 HA 服务配置的 ASE 服务器名称。
Sybase 主目录	**sybase_home**	Sybase 产品的主目录。
登录文件	**login_file**	包含登录密码对的登录文件全路径。
接口文件	**interfaces_file**	用来启动/访问 ASE 服务器的接口文件全路径。
SYBASE_ASE 目录名	**sybase_ase**	sybase_home 中安装 ASE 产品的目录名。
SYBASE_OCS 目录名	**sybase_ocs**	sybase_home 中安装 OCS 产品的目录名。例如：ASE-15_0。

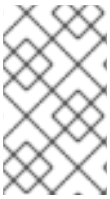
luci 字段	cluster.conf 属性	描述
Sybase 用户	sybase_user	可运行 ASE 服务器的用户。
关闭等待（秒为单位）	start_timeout	启动超时值。
关机等待（秒为单位）	shutdown_timeout	关机超时值。
深度探测超时	deep_probe_timeout	在运行深度探测时，确定服务器没有响应前等待 ASE 服务器响应的最长秒数。

表 B.23. Tomcat 6

luci 字段	cluster.conf 属性	描述
名称	name	指定文件系统名称用于日志记录或者其它目的。
配置文件	config_file	指定配置文件的绝对路径。默认值为 /etc/tomcat6/tomcat6.conf。
关闭等待（秒为单位）	shutdown_wait	指定正常结束关闭服务的要等待的秒数。默认值为 30。

重要

根据 表 B.24 “虚拟机”，当使用虚拟机资源配置集群时，应该使用 **rgmanager** 启动和停止虚拟机。使用 **virsh** 启动虚拟机可导致虚拟机在一个以上的位置中运行，从而造成虚拟机中的数据崩溃。有关配置您的系统，以减少管理员意外使用集群和非集群工具，“重复启动”虚拟机的几率的详情请参考 第 2.14 节 “在集群的环境中配置虚拟机”。



注意

虚拟机资源与其他集群资源配置不同。要使用 **luci** 配置虚拟机资源，请在集群中添加服务组，然后在该服务中添加资源，选择**虚拟机**作为资源类型，并输入虚拟机资源参数。有关使用 **ccs** 配置虚拟机的详情请参考 第 5.12 节 “虚拟机资源”。

表 B.24. 虚拟机

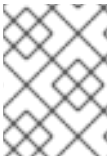
luci 字段	cluster.conf 属性	描述
服务名称	name	指定虚拟机名称。当使用 **luci** 界面时，您可以将其指定为服务名。

luci 字段	cluster.conf 属性	描述
自动启动这个服务	**autostart**	如果启用，则这台虚拟机会在集群达到定额数后自动启动。如果*禁用*这个参数，则这台虚拟机就*不会*在集群达到定额数后自动启动。虚拟机处于 ***disabled*** 状态。
独家运行	**exclusive**	如果启用，这台虚拟机则只能重新定位独占另一个节点运行，即在没有其它虚拟机运行的节点中运行。如果没有可用的虚拟机供其独占运行，则虚拟机在失败后就无法重启。另外，由于使用 ***Run exclusive***，其它虚拟机也无法重新定位到运行这台虚拟机的节点。您可以使用手动启动或者重新定位操作覆盖这个选项。
故障切换域	**domain**	定义在虚拟机失败事件中可尝试的集群成员列表。
恢复策略	**recovery**	***Recovery policy*** 提供以下选项： • ***Disable*** — 失败后禁用该虚拟机。 • ***Relocate*** — 尝试在另一个节点中重启该虚拟机，即不在当前节点中重启。 • ***Restart*** — 在尝试将该虚拟机重新定位（默认选项）到另一个节点中前，尝试在本地（当前节点中）重启该虚拟机。 • ***Restart-Disable*** — 将在服务失败的地方重启该服务。但如果重启该服务失败，则会禁用该服务，而不是将其移动到该集群的另一台主机中。
重启选项	**max_restarts**, **restart_expire_time**	如果您选择**「重启」**或者**「禁用重启」**作为该服务的恢复策略，您可以指定在重新定位或者禁用该服务前最多重启失败的次数，并指定多少秒后不再重启。
迁移类型	**migrate**	指定迁移类型 ***live*** 或者 ***pause***，默认设置为 ***live***。
迁移映射	**migration_mapping**	为迁移指定可替换接口。例如：您可以在当某个节点用于虚拟机迁移的网络地址与该节点用来进行集群通信的地址不同时指定可替换接口。 指定以下说明当您将虚拟机从 **member** 迁移到 **member2** 时，您实际上是迁移到 **target2**。同样，当您从 **member2** 迁移到 **member** 时，您使用 **target** 迁移。 **member:target,member2:target2**
状态程序	**status_program**	除对出现的虚拟机进行常规检查外要运行的状态程序。如果指定，则每分钟执行一次状态程序。这可让您了解虚拟机中关键服务的状态。例如：如果某台虚拟机运行网页服务器，您的状态程序可查看该网页服务器是否启动并运行。如果该状态检查失败（返回一个非零值），就是覆盖了该虚拟机。 启动虚拟机后，该虚拟机资源代理会周期性调用该状态程序，并期待得到一个成功返回代码（0）。5 分钟后超时。

luci 字段	cluster.conf 属性	描述
用来生成虚拟机的 xmlfile 路径	**xmlfile**	到 **libvirt** XML 文件的完整路径包含 **libvirt** 域定义。
虚拟机配置文件路径	**path**	一组用冒号分隔的路径，虚拟机资源代理（**vm.sh**）用它搜索虚拟机配置文件。例如：**/mnt/guests/config:/etc/libvirt/qemu**。  **重要** 该路径应该*永远*不直接指向某个虚拟机配置文件。
虚拟机快照目录路径	**snapshot**	到保存虚拟机映像的快照目录的路径。
Hypervisor URI	**hypervisor_uri**	Hypervisor URI（通常为自动）。
迁移 URI	**migration_uri**	迁移 URI（通常为自动）。
在迁移过程中使用 ssh 传递数据	**tunnelled**	在迁移过程中使用 ssh 传递数据。

附录 C. HA 资源行为

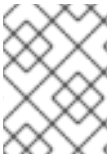
本附录描述了 HA 资源的一般行为，其目的是提供在配置 HA 服务时可能有所帮助的辅助信息。您可使用 **luci** 或者 **/etc/cluster/cluster.conf** 配置参数。有关 HA 资源参数的描述，请参考 附录 B, *HA 资源参数*。要了解资源代理详情请查看集群节点中的 **/usr/share/cluster** 文件。



注意

要完全理解本附录中的信息，您需要对资源代理以及集群配置文件 **/etc/cluster/cluster.conf** 有深入的理解。

HA 服务是一组在统一实体中配置的集群资源，可为客户端提供指定的服务。HA 服务在集群配置文件 **/etc/cluster/cluster.conf**（在每个集群节点中）中以资源树的形式出现。在集群配置文件中，每个资源树都使用一个 XML 代表，它指定每个资源及其属性，以及在资源树中与其他资源的关系（上级、下级和平级关系）。



注意

因为 HA 服务由分为层次树的资源组成，服务有时也指的是*资源树*或者*资源组*。这两个词组与 *HA 服务*有相同的含义。

在每个资源树的顶端是一个特殊的资源类型 — *服务资源*。其他资源类型构成服务的剩余部分，决定服务的特点。配置 HA 服务包括创建服务资源、创建下级集群资源以及将其组成统一实体，形成该服务的分级限制。

本附录由以下小节组成：

- 第 C.1 节 “资源间的上级、下级和同级关系”
- 第 C.2 节 “同级资源启动顺序以及下级资源顺序”
- 第 C.3 节 “继承、<资源>块以及重复使用资源”
- 第 C.4 节 “故障恢复和独立子树”
- 第 C.5 节 “调整并测试服务和资源顺序”

注意

在集群配置文件 **/etc/cluster/cluster.conf** 示例之后的这部分只作为演示使用。

C.1. 资源间的上级、下级和同级关系

集群服务是一个在 **rgmanager** 控制下运行的整合实体。服务中的所有资源都在同一节点中运行。从 **rgmanager** 角度来看，一个集群服务就是一个可启动、停止或者重新定位的实体。但在集群服务中，资源结构决定每个启动和停止的顺序。结构等级包括上级、下级和同级。

例 C.1 “服务 foo 的资源结构” 显示服务 *foo* 的资源树示例。在这个示例中，资源间的关系如下：

- **fs:myfs** (<fs name="myfs" ...>) 和 **ip:10.1.1.2** (<ip address="10.1.1.2 .../>) 是同级。
- **fs:myfs** (<fs name="myfs" ...>) 是 **script:script_child** (<script name="script_child"/>) 的上级。

- **`script:script_child`** (<script name="script_child"/>) 是 **`fs:myfs`** (<fs name="myfs" ...>) 的下级。

例 C.1. 服务 foo 的资源结构

```
<service name="foo" ...>
    <fs name="myfs" ...>
        <script name="script_child"/>
    </fs>
    <ip address="10.1.1.2" .../>
</service>
```

在资源树的上/下级关系中采用以下规则：

- 上级资源在下级资源之前启动。
- 在停止上级资源前必须停止全部下级资源。
- 对于正常工作的资源，其下级资源必须全部正常工作。

C.2. 同级资源启动顺序以及下级资源顺序

服务资源根据是否为子资源指定子类型属性决定子资源的启动和停止顺序，如下：

- 指定子类型属性（*归类的*子资源）— 如果服务资源为子资源指定子类型属性，则该子资源就被*归类*了。该子类型属性明确决定该子资源的启动和停止顺序。
- *不指定*子类型属性（*不归类*子资源）— 如果服务资源*不为*子资源指定子类型属性，则该子资源是*不归类*的。该服务资源不会明确控制不归类子资源的启动和停止顺序。但不归类子资源根据其在 **`/etc/cluster/cluster.conf`** 中的顺序启动和停止。另外，不归类子资源在所有归类子资源启动后启动，并在所有归类子资源停止前停止。

注意

使用定义的*子资源类型*排序的唯一资源是服务资源。

有关归类子资源启动和停止顺序的详情请参考 第 C.2.1 节 “归类子资源启动和停止顺序”。有关不归类子资源启动和停止顺序的详情请参考 第 C.2.2 节 “不归类子资源启动和停止顺序”。

C.2.1. 归类子资源启动和停止顺序

在归类子资源中，子资源的类型属性定义每个资源类型的启动和停止顺序，数字从 1 到 100，一个数值用于启动顺序，一个数值用于停止顺序。数字越小，越早启动或者停止该资源类型。例如：在 表 C.1 “子资源类型启动和停止顺序” 中演示的每个资源类型值；例 C.2 “资源启动和停止值：服务资源代理 **`service.sh`** 除外” 中演示的在服务资源代理 **`service.sh`** 中显示的启动和停止值。在服务资源中，首先启动所有 LVM 子资源，然后是所有文件系统子资源，之后是所有脚本子资源，依此类推。

表 C.1. 子资源类型启动和停止顺序

资源	子类型	启动顺序值	停止顺序值
LVM	lvm	1	9
文件系统	fs	2	8
GFS2 文件系统	clusterfs	3	7
NFS Mount	netfs	4	6
NFS 导出	nfsexport	5	5
NFS 客户端	nfsclient	6	4
IP 地址	ip	7	2
Samba	smb	8	3
脚本	script	9	1

例 C.2. 资源启动和停止值：服务资源代理 `service.sh` 除外

```
<special tag="rgmanager">
    <attributes root="1" maxinstances="1"/>
    <child type="lvm" start="1" stop="9"/>
    <child type="fs" start="2" stop="8"/>
    <child type="clusterfs" start="3" stop="7"/>
    <child type="netfs" start="4" stop="6"/>
    <child type="nfsexport" start="5" stop="5"/>
    <child type="nfsclient" start="6" stop="4"/>
    <child type="ip" start="7" stop="2"/>
    <child type="smb" start="8" stop="3"/>
    <child type="script" start="9" stop="1"/>
</special>
```

资源类型的顺序与其在集群配置文件 **`/etc/cluster/cluster.conf`** 中保留的顺序一致。例如：将其视为 例 C.3 “资源类型中的排序” 中的归类子资源启动和停止顺序。

例 C.3. 资源类型中的排序

```
<service name="foo">
  <script name="1" .../>
  <lvm name="1" .../>
  <ip address="10.1.1.1" .../>
  <fs name="1" .../>
  <lvm name="2" .../>
</service>
```

归类子资源的启动顺序

在 例 C.3 “资源类型中的排序” 中，资源按如下顺序启动：

1. **lvm:1** — 这是 LVM 资源。首先启动所有 LVM 资源。**lvm:1**（**<lvm name="1" .../>**）是 LVM 资源中第一个启动的 LVM 资源，因为它是 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分的第一个 LVM 资源。

2. **lvm:2** — 这是 LVM 资源。首先启动所有 LVM 资源。**lvm:2**（**<lvm name="2" .../>**）是在 **lvm:1** 之后启动的资源，因为它列在 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分 **lvm:1** 之后。

3. **fs:1** — 这是文件系统资源。如果在 Service *foo* 中还有其它文件系统资源，则应按照 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分的顺序启动。

4. **ip:10.1.1.1** — 这是 IP 地址资源。如果在 Service *foo* 中还有其它 IP 地址资源，则应按照 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分的顺序启动。

5. **script:1** — 这是脚本资源。如果在 Service *foo* 中还有其它脚本资源，则应按照 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分的顺序启动。

归类的子资源停止顺序

在 例 C.3 “资源类型中的排序” 中资源按照如下顺序停止：

1. **script:1** — 这是脚本资源。如果在 Service *foo* 中还有其它脚本资源，则应按照与 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分相反的顺序停止。

2. **ip:10.1.1.1** — 这是 IP 地址资源。如果在 Service *foo* 中还有其它 IP 地址资源，则应按照与 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分相反的顺序停止。

3. **fs:1** — 这是文件系统资源。如果在 Service *foo* 中还有其它文件系统资源，则应按照与 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分相反的顺序停止。

4. **lvm:2** — 这是 LVM 资源。最后停止所有 LVM 资源。**lvm:2**（**<lvm name="2" .../>**）是在 **lvm:1** 之前停止的资源，资源类型组中的资源按照 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分的相反顺序停止。

5. **lvm:1** — 这是 LVM 资源。最后停止所有 LVM 资源。**lvm:1**（**<lvm name="1" .../>**）是在 **lvm:2** 之后停止的资源，资源类型组中的资源按照 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分的相反顺序停止。

C.2.2. 不归类子资源启动和停止顺序

未归类的子资源需要额外的注意事项。对于未归类子资源，该服务资源没有明确指定启动顺序和停止顺序。反之要根据子资源在 **/etc/cluster/cluster.conf** 中的顺序决定启动顺序和停止顺序。另外，未归类子资源在停止所有归类子资源后启动，并在停止所有归类子资源前停止。

例如：在 例 C.4 “服务中的不归类和归类子资源” 中演示的不归类子资源启动和停止顺序。

例 C.4. 服务中的不归类和归类子资源

```
<service name="foo">
  <script name="1" .../>
  <nontypedresource name="foo"/>
```

```
    <lvm name="1" .../>
    <nontypedresourcetwo name="bar"/>
    <ip address="10.1.1.1" .../>
    <fs name="1" .../>
    <lvm name="2" .../>
  </service>
```

不归类子资源启动顺序

在 例 C.4 “服务中的不归类和归类子资源” 中子资源按如下顺序启动：

1. **lvm:1** — 这是 LVM 资源。首先启动所有 LVM 资源。**lvm:1**（**<lvm name="1" .../>**）是 LVM 资源中第一个启动的 LVM 资源，因为它是 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分的第一个 LVM 资源。

2. **lvm:2** — 这是 LVM 资源。首先启动所有 LVM 资源。**lvm:2**（**<lvm name="2" .../>**）是在 **lvm:1** 之后启动的资源，因为它列在 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分 **lvm:1** 之后。

3. **fs:1** — 这是文件系统资源。如果在 Service *foo* 中还有其它文件系统资源，则应按照 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分的顺序启动。

4. **ip:10.1.1.1** — 这是 IP 地址资源。如果在 Service *foo* 中还有其它 IP 地址资源，则应按照 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分的顺序启动。

5. **script:1** — 这是脚本资源。如果在 Service *foo* 中还有其它脚本资源，则应按照 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分的顺序启动。

6. **nontypedresource:foo** — 这是不归类资源。因为它是不归类资源，因此在归类资源之后启动。另外，它在服务资源中的顺序在其它不归类资源 **nontypedresourcetwo:bar** 前面，因此它在 **nontypedresourcetwo:bar** 之前启动。（不归类资源按照它们在服务资源中出现的顺序启动。）

7. **nontypedresourcetwo:bar** — 这是不归类资源。因为它是不归类资源，因此在归类资源之后启动。另外，它在服务资源中的顺序在其它不归类资源 **nontypedresourcetwo:foo** 后面，因此它在 **nontypedresourcetwo:foo** 之后启动。（不归类资源按照它们在服务资源中出现的顺序启动。）

不归类资源停止顺序

在 例 C.4 “服务中的不归类和归类子资源” 中子资源按照如下顺序停止：

1. **nontypedresourcetwo:bar** — 这是不归类资源。因为它是不归类资源，因此在归类资源之前停止。另外，它在服务资源中的顺序在其它不归类资源 **nontypedresourcetwo:foo** 后面，因此它在 **nontypedresourcetwo:foo** 之前停止。（不归类资源按照它们在服务资源中出现的相反顺序停止。）

2. **nontypedresource:foo** — 这是不归类资源。因为它是不归类资源，因此在归类资源之前停止。另外，它在服务资源中的顺序在其它不归类资源 **nontypedresourcetwo:bar** 前面，因此它在 **nontypedresourcetwo:bar** 之后停止。（不归类资源以其在服务资源中出现的相反顺序停止。）

3. **script:1** — 这是脚本资源。如果在 Service *foo* 中还有其它脚本资源，则应按照与 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分相反的顺序停止。

4. **ip:10.1.1.1** — 这是 IP 地址资源。如果在 Service *foo* 中还有其它 IP 地址资源，则应按照与 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分相反的顺序停止。

5. **fs:1** — 这是文件系统资源。如果在 Service *foo* 中还有其它文件系统资源，则应按照与 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分相反的顺序停止。

6. **lvm:2** — 这是 LVM 资源。最后停止所有 LVM 资源。**lvm:2**（**<lvm name="2" .../>**）是在 **lvm:1** 之前停止的资源，资源类型组中的资源按照 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分的相反顺序停止。

7. **lvm:1** — 这是 LVM 资源。最后停止所有 LVM 资源。**lvm:1**（**<lvm name="1" .../>**）是在 **lvm:2** 之后停止的资源，资源类型组中的资源按照 **/etc/cluster/cluster.conf** 文件 Service *foo* 部分的相反顺序停止。

C.3. 继承、<资源>块以及重复使用资源

有些资源利用从上级资源中继承的值，通常是在 NFS 服务中。例 C.5 “资源再利用和继承的 NFS 服务设置” 演示了典型的 NFS 服务配置，资源再利用和继承设置。

例 C.5. 资源再利用和继承的 NFS 服务设置

```
    <resources>
        <nfsclient name="bob" target="bob.example.com"
options="rw,no_root_squash"/>
        <nfsclient name="jim" target="jim.example.com"
options="rw,no_root_squash"/>
        <nfsexport name="exports"/>
    </resources>
    <service name="foo">
        <fs name="1" mountpoint="/mnt/foo" device="/dev/sdb1"
fsid="12344">
            <nfsexport ref="exports">  <!-- nfsexport's path and fsid
attributes
                                           are inherited from the
mountpoint &
                                           fsid attribute of the
parent fs
                                           resource -->
                <nfsclient ref="bob"/> <!-- nfsclient's path is
inherited from the
                                           mountpoint and the fsid
is added to the
                                           options string during
export -->
                <nfsclient ref="jim"/>
            </nfsexport>
        </fs>
        <fs name="2" mountpoint="/mnt/bar" device="/dev/sdb2"
fsid="12345">
            <nfsexport ref="exports">
                <nfsclient ref="bob"/> <!-- Because all of the critical
```

```
data for this
                                                    resource is either
defined in the
                                                    resources block or
inherited, we can
                                                    reference it again! -->
                <nfsclient ref="jim"/>
            </nfsexport>
        </fs>
        <ip address="10.2.13.20"/>
    </service>
```

如果服务是平面的（即没有上级/下级关系），则需要按使用如下配置：

- 该服务需要四个 nfsclient 资源 — 每个文件系统一个（一共两个文件系统），每个目标机器一个（一共两台目标机器）。
- 服务需要为每个 nfsclient 指定 export path 和 file system ID，配置的这个部分容易出错。

但在 例 C.5 “资源再利用和继承的 NFS 服务设置” 中只定义一次 NFS 客户端资源 *nfsclient:bob* 和 *nfsclient:jim*；同样也只定义一次 NFS 导出资源 *nfsexport:exports*。资源需要的所有属性都是从上级资源中继承的。因为继承的属性是动态的（并不与其它属性冲突），因此可能再利用那些资源 — 也就是为什么要在资源块中定义它们的原因。在很多情况下可能对配置一些资源没有可操作性。例如：在很多情况下配置文件系统资源可导致在两个节点中挂载一个文件系统，这样就会出问题。

C.4. 故障恢复和独立子树

在很多企业级环境中，如果服务的组件失败，修复的正常动作是重启整个服务。例如：在 例 C.6 “服务 *foo* 常见故障恢复” 中，如果这个服务中定义的任意脚本失败，则正常动作是重启（重新定位或者禁用，视服务恢复策略而定）该服务。但在有些情况下，服务的某些部分可能是不重要的，可能有必要在尝试一般恢复前重启该服务的某一部分。要完成这个操作，您可以使用 ***__independent_subtree*** 属性。例如：在 例 C.7 “使用 ***__independent_subtree*** 属性对服务 *foo* 执行故障恢复” 中，可使用 ***__independent_subtree*** 属性完成以下动作：

- 如果 script:script_one 失败，重启 script:script_one、script:script_two 和 script:script_three。
- 如果 script:script_two 失败，则只重启 script:script_two。
- 如果 script:script_three 失败，则重启 script:script_one、script:script_two 和 script:script_three。
- 如果 script:script_four 失败，则重启整个服务。

例 C.6. 服务 *foo* 常见故障恢复

```
<service name="foo">
      <script name="script_one" ...>
          <script name="script_two" .../>
      </script>
      <script name="script_three" .../>
</service>
```

例 C.7. 使用 *__independent_subtree* 属性对服务 *foo* 执行故障恢复

```
<service name="foo">
      <script name="script_one" __independent_subtree="1" ...>
          <script name="script_two" __independent_subtree="1" .../>
          <script name="script_three" .../>
      </script>
      <script name="script_four" .../>
</service>
```

在有些情况下，如果服务的某个组件失败，您想要只禁用那个组件而不禁用整个服务，以避免影响使用那个服务其他组件的服务。从红帽企业版 Linux 6.1 发行本开始，您可以使用 ***__independent_subtree="2"*** 属性达到此目的，该属性将独立子树标为非关键（non-critical）。

注意

您可以在单一参考的资源中只使用 non-critical 标签。这个 non-critical 标签可在所有资源中使用，并可用于资源树的所有层，但在定义服务或者虚拟机时不应在顶层使用。

从红帽企业版 Linux 6.1 开始，您可以在该资源数的每个节点中为独立的子树设定最多重启和重启过期。您可以使用 以下属性设定这些阈值：

- **__max_restarts** 设定在放弃努力前最多可承受的重启次数。
- **__restart_expire_time** 以秒为单位设定在多长时间后不再尝试重启。

C.5. 调整并测试服务和资源顺序

您可以使用 **rg_test** 程序调整并测试服务和资源顺序。**rg_test** 是命令行工具，它由 **rgmanager** 软件包提供，在 shell 或者终端中运行（在 **Conga** 中不可用）。表 C.2 “**rg_test** 程序小结” 中总结了 **rg_test** 的动作和语法。

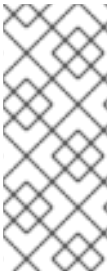
表 C.2. rg_test 程序小结

动作	语法
显示 **rg_test** 理解的资源规则。	**rg_test rules**
测试配置（和 /usr/share/cluster）中的错误和冗余资源代理。	**rg_test test /etc/cluster/cluster.conf**

动作	语法
显示服务的启动和停止排序。	显示启动顺序： **rg_test noop /etc/cluster/cluster.conf start service *servicename*** 显示停止顺序： **rg_test noop /etc/cluster/cluster.conf stop service *servicename***
明确启动或者停止服务。	 **重要** 只在一个节点中这样做并总是先禁用 rgmanager 中的服务。 启动服务： **rg_test test /etc/cluster/cluster.conf start service *servicename*** 停止服务： **rg_test test /etc/cluster/cluster.conf stop service *servicename***
计算并显示两个 cluster.conf 文件的资源树 delta。	**rg_test delta *cluster.conf file 1 cluster.conf file 2*** 例如： **rg_test delta /etc/cluster/cluster.conf.bak /etc/cluster/cluster.conf**

附录 D. 集群服务资源检查及故障切换超时

本附录论述了 **rgmanager** 如何监控集群资源状态，以及如何修改状态检查间隔。本附录还论述了 **__enforce_timeouts** 服务参数，它说明操作超时可造成服务失败。



注意

要完整理解本附录中的信息，您需要对资源代理和集群配置文件 **/etc/cluster/cluster.conf** 有深入的了解。有关 **cluster.conf** 元素和属性的完整列表，请参考 **/usr/share/cluster/cluster.rng** 中的集群方案，注释的方案位于 **/usr/share/doc/cman-X.Y.ZZ/cluster_conf.html**（例如：**/usr/share/doc/cman-3.0.12/cluster_conf.html**）。

D.1. 修改资源状态检查间隔

rgmanager 检查各个资源的状态，而不是整个服务。rgmanager 每 10 秒钟扫描一次资源树，检查那些超过其“状态检查”间隔的资源。

每个资源代理都指定状态检查周期间的时间。每个资源都使用这些超时值，除非在 **cluster.conf** 中使用特殊 **<action>** 标签明确说明。

```
<action name="status" depth="*" interval="10" />
```

这个标签是 **cluster.conf** 文件中资源本身衍生的文件系统。例如：如果您有一个文件系统，您想要覆盖其状态检查间隔，您就可以在 **cluster.conf** 文件中按如下方法指定文件系统：

```
<fs name="test" device="/dev/sdb3">
  <action name="status" depth="*" interval="10" />
  <nfsexport...>
  </nfsexport>
</fs>
```

有些代理提供多个检查“深度”。例如：常规文件系统状态检查（深度 0）检查该文件系统是否挂载到正确的位置。更集中的检查（深度 10）则会查看您是否可以从文件系统中读取文件。状态检查深度 20 会查看您是否可以写入该文件系统。在给出的示例中将 **depth** 设定为 *****，表示这些值可用于所有深度。结果是每 10 秒钟对 **test** 文件系统执行一次由资源代理提供的最高定义的深度（在此是 20）检查 。

D.2. 强制资源超时

启动、停止或故障切换资源时没有超时。有些资源的启动或停止时间不确定。遗憾的是无法停止（包括超时）会造成服务不可操作（失败的状态）。如果需要，您可以在各个服务的每个资源中打开超时强制，方法是在 **cluster.conf** 文件中添加 **__enforce_timeouts="1"** 参考。

以下示例演示了使用 **__enforce_timeouts** 属性为 **netfs** 资源设置的集群服务。使用这个设置，如果在恢复过程中，超过 30 秒还没有卸载 NFS 文件系统，则操作会超时，该服务进入失败状态。

```
</screen>
<rm>
  <failoverdomains/>
  <resources>
```

```
    <netfs export="/nfstest" force_unmount="1" fstype="nfs"
host="10.65.48.65"
           mountpoint="/data/nfstest" name="nfstest_data"
options="rw,sync,soft"/>
  </resources>
  <service autostart="1" exclusive="0" name="nfs_client_test"
recovery="relocate">
    <netfs ref="nfstest_data" __enforce_timeouts="1"/>
  </service>
</rm>
```

附录 E. 命令行工具小结

表 E.1 “命令行工具小结” 总结了配置和管理高可用性附加组件的首选命令行工具。有关命令和变量的详情请参考每个命令行工具的 man page。

表 E.1. 命令行工具小结

命令行工具	使用	目的
ccs_config_dump — 集群配置转储工具	集群基础设施	**ccs_config_dump** 生成正在运行配置的 XML 输出。运行的配置有时与文件中保存的配置有所不同，因为有些子系统在配置中保存或者设定一些默认信息。那些值通常不出现在配置的磁盘版本中，但在运行时需要它们方可使集群正常工作。
ccs_config_validate — 集群配置验证工具	集群基础设施	**ccs_config_validate** 根据方案 **cluster.rng**（在每个节点中位于 **/usr/share/cluster/cluster.rng**）验证 **cluster.conf**。有关这个工具的详情请参考 ccs_config_validate(8) man page。
clustat — 集群状态工具	高可用性服务管理组件	**clustat** 命令显示集群的状态。它显示成员信息、仲裁查看以及所有配置的用户服务状态。有关这个工具的详情请参考 clustat(8) man page。
clusvcadm — 集群用户服务管理工具	高可用性服务管理组件	**clusvcadm** 命令允许您在集群中启用、禁用、重新定位以及重启高可用性服务。有关这个工具的详情请参考 clusvcadm(8) man page。
cman_tool — 集群管理工具	集群基础设施	**cman_tool** 是管理 CMAN 集群管理器的程序。它可提供加入或者离开集群、杀死节点或者更改集群中节点预期仲裁投票的功能。有关这个工具的详情请参考 cman_tool(8) man page。
fence_tool — Fence 工具	集群基础设施	**fence_tool** 是用来加入和离开 fence 域的程序。有关这个工具的详情请参考 fence_tool(8) man page。

附录 F. 高可用性 LVM（HA-LVM）

红帽高可用性附加组件在故障切换配置中支持高可用性 LVM 卷（HA-LVM）。这与集群的逻辑卷管理器（CLVM）启用的 active/active 配置不同，它是 LVM 的一组集群扩展，可让计算机集群管理共享存储。

应根据所部署应用程序或服务的需要使用 CLVM 或 HA-LVM。

- 如果应用程序是在全局可识别，并已调整为每次同时在多台机器中运行，那就应该使用 CLVM。特别是如果您集群中一个以上的节点要求访问在活动节点间共享的存储，您就必须使用 CLVM。CLVM 允许用户在共享存储中配置逻辑卷，方法是配置逻辑卷时锁定对物理存储的访问，并使用集群的锁定服务管理共享存储。有关 CLVM 及 LVM 常规配置详情请参考 *《管理逻辑卷管理器》*。

- 如果应用程序以最佳的 active/passive（故障切换）配置运行，那么一次只有一个访问该存储的单一节点是活动的，您就可以使用高可用逻辑卷管理代理（HA-LVM）。

大多数应用程序在 active/passive 配置中运行更佳，因为它们不是设计或者优化以便与其他事务同时运行。如果逻辑卷是镜像的，那么选择运行一个在集群的逻辑卷中无法识别的应用程序可能导致性能下降。这是因为集群沟通会消耗这些事务中的逻辑卷。在集群中可识别的应用程序必须可以让获得的性能比集群文件系统和集群可识别逻辑卷损失的性能多。有些应用程序和工作负载更容易达到此目的。确定集群要求是什么，以及在 active/active 集群优化时的额外努力是否有益才是在两种 LVM 变体间进行选择的依据。大多数用户将会从 HA-LVM 中获得最佳 HA 效果。

HA-LVM 和 CLVM 相似，它们都可以防止 LVM 元数据及其逻辑卷崩溃；反之，如果允许多台机器执行互相可覆盖的更改，则会发生数据和逻辑卷崩溃。HA-LVM 强制限制只能单独激活逻辑卷，即一次只能在一台机器中激活。这就是说只能使用存储驱动器的本地（非集群）部署。避免这种形式的集群合作可提高性能。CLVM 没有这些强制限制，用户可以随意在集群的所有机器中激活逻辑卷。这样就强制使用集群可识别存储驱动器，即允许将集群可识别文件系统和应用程序放在顶层。

可将 HA-LVM 设置为使用两种方法之一达到其强制独家逻辑卷激活的目的。

- 首选方法是使用 CLVM，但它只能激活唯一的逻辑卷。好处是可轻松设置并有效防止管理失误（比如删除正在使用的逻辑卷）。要使用 CLVM，则必须运行高可用性附加组件软件和弹性存储附加组件软件，包括 **clvmd**。

  使用这个方法配置 HA-LVM 的步骤请参考 第 F.1 节 “使用 CLVM 配置 HA-LVM 故障切换（首选）”。

- 第二种方法使用本地机器锁定和 LVM“标签”。这个方法的优点是不需要任何 LVM 集群软件包，但-设置步骤比较复杂，且无法防止管理员意外从不活动的集群中删除逻辑卷。使用这个方法配置 HA-LVM 的步骤请参考 第 F.2 节 “使用标签配置 HA-LVM 故障切换”。

F.1. 使用 CLVM 配置 HA-LVM 故障切换（首选）

要设置 HA-LVM 故障切换（使用首选 CLVM 变体），请执行以下步骤：

1. 确定将您的系统配置为支持 CLVM，要求如下：

   - 如果 CLVM 逻辑卷不是镜像的，请安装高可用性附加组件和弹性存储附加组件，包括 **cmirror** 软件包。

   - 请将 **/etc/lvm/lvm.conf** 文件 global 部分的 **locking_type** 参数设定为 ‘3’。

   - 必须运行高可用性附加组件和弹性存储附加组件，包括 **cmirror** 软件包。在 CLVM 镜像中，还必须启动 **cmirrord** 服务。

2. 使用标准 LVM 和文件系统命令生成逻辑卷和文件系统，如以下示例所示：

```
# pvcreate /dev/sd[cde]1

# vgcreate -cy shared_vg /dev/sd[cde]1

# lvcreate -L 10G -n ha_lv shared_vg

# mkfs.ext4 /dev/shared_vg/ha_lv

# lvchange -an shared_vg/ha_lv
```

有关生成 LVM 逻辑卷的详情请参考*《管理逻辑卷过滤器》*。

3. 编辑 **/etc/cluster/cluster.conf** 文件，使其包含新生成的逻辑卷作为您服务之一的资源。另外，您可以使用 **Conga** 或者 **ccs** 命令为集群配置 LVM 和文件系统资源。以下是 **/etc/cluster/cluster.conf** 文件中将 CLVM 逻辑卷配置为集群资源的资源管理器部分示例：

```
<rm>
   <failoverdomains>
       <failoverdomain name="FD" ordered="1" restricted="0">
          <failoverdomainnode name="neo-01" priority="1"/>
          <failoverdomainnode name="neo-02" priority="2"/>
       </failoverdomain>
   </failoverdomains>
   <resources>
       <lvm name="lvm" vg_name="shared_vg" lv_name="ha-lv"/>
       <fs name="FS" device="/dev/shared_vg/ha-lv" force_fsck="0"
force_unmount="1" fsid="64050" fstype="ext4" mountpoint="/mnt"
options="" self_fence="0"/>
   </resources>
   <service autostart="1" domain="FD" name="serv"
recovery="relocate">
       <lvm ref="lvm"/>
       <fs ref="FS"/>
   </service>
</rm>
```

F.2. 使用标签配置 HA-LVM 故障切换

要使用标签在 **/etc/lvm/lvm.conf** 文件中设置 HA-LVM 故障切换，请执行以下步骤：

1. 确定将 **/etc/lvm/lvm.conf** 文件 global 部分的 **locking_type** 参数设定为 '1'：

2. 使用标准 LVM 和文件系统命令生成逻辑卷和文件系统，如以下示例所示：

```
# pvcreate /dev/sd[cde]1

# vgcreate shared_vg /dev/sd[cde]1
```

```
# lvcreate -L 10G -n ha_lv shared_vg

# mkfs.ext4 /dev/shared_vg/ha_lv
```

有关生成 LVM 逻辑卷的详情请参考*《管理逻辑卷过滤器》*。

3. 编辑 **/etc/cluster/cluster.conf** 文件，使其包含新生成的逻辑卷作为您服务之一的资源。另外，您可以使用 **Conga** 或者 **ccs** 命令为集群配置 LVM 和文件系统资源。以下是 **/etc/cluster/cluster.conf** 文件中将 CLVM 逻辑卷配置为集群资源的资源管理器部分示例：

```
<rm>
   <failoverdomains>
       <failoverdomain name="FD" ordered="1" restricted="0">
          <failoverdomainnode name="neo-01" priority="1"/>
          <failoverdomainnode name="neo-02" priority="2"/>
       </failoverdomain>
   </failoverdomains>
   <resources>
       <lvm name="lvm" vg_name="shared_vg" lv_name="ha_lv"/>
       <fs name="FS" device="/dev/shared_vg/ha_lv" force_fsck="0"
force_unmount="1" fsid="64050" fstype="ext4" mountpoint="/mnt"
options="" self_fence="0"/>
   </resources>
   <service autostart="1" domain="FD" name="serv"
recovery="relocate">
       <lvm ref="lvm"/>
       <fs ref="FS"/>
   </service>
</rm>
```

注意

如果在卷组中有多个逻辑卷，那么 **lv_name** 资源中的逻辑卷名称（**lv_name**）为空白或未指定。另外请注意在 HA-LVM 配置中，一个卷组只能被单一服务使用。

4. 编辑 **/etc/lvm/lvm.conf** 文件的 **volume_list** 字段，使其包括您的 root 卷组名称，同时使用 @ 覆盖在 **/etc/cluster/cluster.conf** 文件中列出的主机名。这里的主机名是您在其中编辑 **lvm.conf** 文件的机器名称。注：这个字符串*必须*与 **cluster.conf** 文件中给出的节点名称映射。下面是 **/etc/lvm/lvm.conf** 文件中的条目示例：

```
volume_list = [ "VolGroup00", "@neo-01" ]
```

这个标签将用来激活共享卷组或逻辑卷。*不要*包括任何要使用 HA-LVM 共享的卷组名称。

5. 在所有集群节点中更新 **initrd** 设备：

```
# dracut -H -f /boot/initramfs-$(uname -r).img $(uname -r)
```

6. 重启所有节点以保证使用正确的 **initrd** 设备。

附录 G. 修订记录

修订 5.0-25.2.400 **2013-10-31** **Rüdiger Landmann**
Rebuild with publican 4.0.0

修订 5.0-25.2 **Wed May 1 2013** **Leah Liu**
完成翻译、校对

修订 5.0-25.1 **Thu Apr 18 2013** **Chester Cheng**
与 XML 源 5.0-25 版本同步的翻译文件

修订 5.0-25 **Mon Feb 18 2013** **Steven Levine**
6.4 GA 发行本

修订 5.0-23 **Wed Jan 30 2013** **Steven Levine**
解决：#901641
修改并说明 IPTables 规则。

修订 5.0-22 **Tue Jan 29 2013** **Steven Levine**
解决：788636
编写 **ccs** 命令 RRP 配置文档。

解决：789010
编写在 **cluster.conf** 文件中进行 RRP 配置。

修订 5.0-20 **Fri Jan 18 2013** **Steven Levine**
解决：894097
删除建议以保证不再使用 VLAN 标签。

解决：845365
说明现在不支持捆绑模式 0 和 2。

修订 5.0-19 **Thu Jan 17 2013** **Steven Levine**
解决：896234
说明集群节点参考术语。

修订 5.0-16 **Mon Nov 26 2012** **Steven Levine**
6.4 Beta 发行本

修订 5.0-15 **Wed Nov 20 2012** **Steven Levine**

解决：838988
编写文件系统资源代理的 nfsrestart 属性。

解决：843169
编写 IBM iPDU fence 代理。

解决：846121
编写 Eaton 网络电源控制器（SNMP 接口）Fence 设备。

解决：856834
编写 HP 刀片系统 fence 代理。

解决：865313
编写 NFS 服务器资源代理。

解决：862281
阐述使用 **ccs** 命令覆盖之前的设置。

解决：846205
编写 **igmp** 组件的 **iptables** 防火墙过滤。

解决：857172
编写如何从 luci 中删除用户。

解决：857165
编写 IPMI fence 代理的特权等级参数。

解决：840912
使用资源参数表清除格式问题。

即将：849240, 870292
说明安装步骤。

解决：871165
说明在 IP 地址资源代理中进行描述的 IP 地址参数。

解决：845333, 869039, 856681
修改一些小的排版错误，并明确一些技术疑点。

修订 5.0-12 **Thu Nov 1 2012** **Steven Levine**
添加新支持的 fence 代理。

修订 5.0-7 **Thu Oct 25 2012** **Steven Levine**
添加新一节覆盖语义。

修订 5.0-6 **Tue Oct 23 2012** **Steven Levine**
修正 Post Join Delay 默认值。

修订 5.0-4 **Tue Oct 16 2012** **Steven Levine**
添加 NFS 服务器资源描述。

修订 5.0-2 **Thu Oct 11 2012** **Steven Levine**
更新 Conga 描述。

修订 5.0-1 **Mon Oct 8 2012** **Steven Levine**
明确 ccs 语义。

修订 4.0-5 **Fri Jun 15 2012** **Steven Levine**

发布版本 6.3 GA

修订 4.0-4 **Tue Jun 12 2012** **Steven Levine**

解决：830148
保证 luci 示例中端口号一致。

修订 4.0-3 **Tue May 21 2012** **Steven Levine**

解决：696897
在 fence 参数和资源参数表格中添加 cluster.conf 参数信息。

解决：811643
添加在独立机器中恢复 **luci** 数据库的步骤。

修订 4.0-2 **Wed Apr 25 2012** **Steven Levine**

解决：815619
删除关于在 GFS2 文件系统中使用 UDP 单播的警告。

修订 4.0-1 **Fri Mar 30 2012** **Steven Levine**

解决：771447、800069、800061
更新 **luci** 文档，使之与红帽企业版 Linux 6.3 保持一致。

解决：712393
添加为 RGManager 捕获应用程序核的信息。

解决：800074
编写 `condor` 资源代理文档。

解决：757904
编写 **luci** 配置备份和恢复文档。

解决：772374
添加在集群中管理虚拟机的一节。

解决：712378
添加 HA-KVM 配置文档。

解决：712400
编写 debug 选项文档。

解决：751156
编写 `fence_ipmilan` 参数文档。

解决：721373
编写那些配置更改需要重启集群的文档。

修订 3.0-5 **Thu Dec 1 2011** **Steven Levine**

红帽企业版 Linux 6.2 的 GA 发行版本

解决：755849
修改 monitor_link 参数示例。

修订 3.0-4 **Mon Nov 7 2011** **Steven Levine**

解决：749857
添加 RHEV-M REST API fence 设备文档。

修订 3.0-3 **Fri Oct 21 2011** **Steven Levine**

解决：#747181, #747182, #747184, #747185, #747186, #747187, #747188, #747189, #747190, #747192
修改在红帽企业版 Linux 6.2 文档 QE 过程中发现的排版错误和模糊之处。

修订 3.0-2 **Fri Oct 7 2011** **Steven Levine**

解决：#743757
修改在故障排除一节中支持的绑定模式参考。

修订 3.0-1 **Wed Sep 28 2011** **Steven Levine**

红帽企业版 Linux 6.2 Beta 发行本初始修订

解决：#739613
编写显示可用 fence 设备和可用服务的新 **ccs** 选项支持文档。

解决：#707740
编写 Conga 界面更新以及对设置用户权限来管理 Conga 的支持文档。

解决：#731856
编写使用 **/etc/sysconfig/luci** 文件配置 **luci** 的支持文档。

解决：#736134
编写 UDPU 传输支持文档。

解决：#736143
编写集群的 Samba 支持文档。

解决：#617434
编写如何配置唯一提供 **luci** 的 IP 地址文档。

解决：#713259
编写 **fence_vmware_soap** 代理支持文档。

解决：#721009
提供支持必需文章的链接。

解决：#717006
提供允许多播流量通过 **iptables** 防火墙的信息。

解决：#717008
提供关于集群服务状态检查和故障切换超时的信息。

解决：#711868
明确自动启动描述。

解决：#728337
编写使用 **ccs** 命令添加 **vm** 资源的步骤文档。

解决：#725315, #733011, #733074, #733689
修改排版错误。

修订 2.0-1 **Thu May 19 2011** **Steven Levine**

红帽企业版 Linux 6.1 发行本初始修订

解决：#671250
编写 SNMP trap 支持文档。

解决：#659753
编写 **ccs** 命令文档。

解决：#665055
更新 Conga 文档，添加更新的显示和功能支持。

解决：#680294
编写 **ricci** 代理的密码访问文档。

解决：#687871
添加故障排除一章。

解决：#673217
修复排版错误。

解决：#675805
在 HA 资源参数表中添加 **cluster.conf** 方案参考。

解决：#672697
更新 fence 设备参数，添加目前支持的所有 fencing 设备。

解决：#677994
修改 **fence_ilo** fence 代理参数信息。

解决：#629471
添加有关在双节点集群中设定 consensus 值的技术说明。

解决：#579585
更新升级红帽高可用性附加组件软件一节的内容。

解决：#643216
阐明文档中的一些小问题。

解决：#643191
改进并修改 **luci** 文档。

解决：#704539
更新虚拟机资源参数表。

修订 1.0-1 **Wed Nov 10 2010** **Paul Kennedy**
红帽企业版 Linux 6 初始发行本

索引

符号

A

B

C

D

S

T

V

W