中标麒麟高可用集群软件（龙芯版）V7.0
产品白皮书

中标软件有限公司

目录

前言

随着社会的发展和整个社会服务意识的增强，越来越多的行业需要提供 7×24 的不间断服务。社会需要一种可以满足这种需求的产品，而中标麒麟高可用集群软件（简称 NeoKylinHA）正是可以满足这种需求的一款企业级高可用性集群解决方案的产品。该产品具有诸多技术特性，其智能的服务切换策略、数据保护和磁盘镜像等功能，确保您的系统数据的可用性、连续性和完整性。

中标麒麟高可用集群软件支持 X86 、X86-64、MIPS64、ARM64、Power 等系列的所有主流 Linux 系统平台，能方便地为您打造一个能满足您的需求又性价比极高的系统环境。

内容指南

NeoKylinHA 是一个基于 Linux 系统平台的集群技术；NeoKylinHA 对网络中的重要系统数据及服务保护具有很高的实用性和易处理性。它是一个支持失效转移、自动恢复及负载均衡的节点集群产品。技术说明是供管理员创建及管理 Linux 高可用集群使用的参考文档。

中标麒麟高可用集群软件产品介绍

欢迎您使用中标麒麟高可用集群软件！

随着信息化浪潮的风起云涌，许多用户正在寻求一种既能支持关键应用运行需要，同时可以降低 IT 基础总体拥有成本的解决之道。Linux 操作系统作为开源软件的重要组成部分，为用户提供了稳定、高效、安全的开放系统平台，使其成为面向网络与数据中心应用的首选平台。

中标麒麟高可用集群软件是基于中标麒麟 Linux 服务器操作系统开发的智能高可用软件产品，通过应用中标麒麟高可用产品可以提升软硬件系统及应用运行的稳定性和可靠性，该产品经过多年的用户应用及市场验证，提供的抗错能力足以支持关键业务系统应用可靠性苛刻要求为政府、金融、电力、医疗、运输、制造业、等行业用户提供高效、至微的可靠服务。中标麒麟高可用集群软件软件依托“系统可靠”—“数据可靠”—“应用可靠”三层优势，实现对业务系统的高可用保护。

智能、多层次保护

■ 系统可靠：保护用户的操作系统及硬件设备，对故障操作系统及硬件设备进行智能、快速、便捷的恢复。

■ 数据可靠：为用户共享数据提供一致性保护，当系统出现脑裂等极端故障的情况下，保证数据不被破坏。

■ 应用可靠：为用户业务系统稳定、高效、持续的运行提供缜密可靠保护。

资源保护

	分类	保护资源
常用应用服务保护	系统网络服务	NFS－SMB
		HTTP
		Sendmail
		DHCP
		IP
		Iptables
		IPVSADM
		Ldap

		DNS
		FTP
		Squid
	商业软件	NeoShine Mail Server
中间件应用保护	国际商业软件	Weblogic
		Webspere
	国内商业软件	Tongweb
		Kingdee
	开源软件	Tomcat
数据库应用保护	国际商业软件	Oracle
		Sybase
		Informix
		DB2
		SAP-DB
	国内商业软件	DM
		Kingbase
		OScard
	开源软件	Mysql
		Postgresql
系统级服务	系统服务	Filesystem
		Linux-scsi
		ISCSI
		RAID
		Software-Raid
		RAW
		LVM
		Sfex
注：中标麒麟高可用集群软件具有良好的 2 次开发接口，可以针对特殊服务，进行定制开发。		

支持环境

服务器：龙芯3A、龙芯3B的CPU国产龙芯CPU架构的主流品牌服务器

操作系统：中标麒麟高级服务器操作系统（龙芯64位）V7.0

存储设备：IBM、EMC、HP和HDS等主流高中低端存储设备

1 概述

1.1 产品特点

NeoKylinHA 包含几个重要的特性能帮助更好的保护和有效的管理着网络数据资源。包括：

持以光纤及 iSCSI 方式为存储的网络。

多节点活跃的集群，由 16 个以上的 Linux 服务器构成。任何在这个集群中的服务器都可以重新提供群中失效服务器的资源（应用软件、服务，IP 住址和文件系统）。

通过图形工具或命令行工具管理集群中的节点，通过双向的工具让您配置和监视高可用集群。

以具体的应用程序和硬件基础设施的性能去适应您集群的组织结构。

根据在用户需求进行服务器储存库的动态分配和指定。

依照时间的配置使资源自动恢复到时间列表中的修复节点。

支持共享磁盘体系，虽然支持共享磁盘体系，但不是必须的。

支持集群文件系统，例如：OCFS2 等。

支持群集识别逻辑卷管理，例如：EVMS 等。

1.2 产品优势

NeoKylinHA 可以让用户设定多达 16 个以上的 Linux 服务器的高可用性集群，其运行资源可以被动态地切换或移动到任何在群集中的服务器。如果在集群中发生一个资源服务器故障，资源可自动切换迁移或手动迁移。

NeoKylinHA 从产品的组成上提供了高可用性。通过更低的代价获得集群上应用系统和操作的完全统一。NeoKylinHA 也可以实现让您集中管理完整的集群和调整资源，以应付不断转变的工作量的要求（从而，手动“负载均衡”集群）。

其重要的好处是减少潜在的无计划的服务中断，以及有计划的停止服务以便于更好的维修和升级服务器的软硬件。实施 NeoKylinHA 高可用集群软件的优势有：

强大的高可用性；

良好的性能；

运转实施的低成本；

可延展性；

系统数据大范围损坏的恢复；

数据保护；

服务器的联合统一。

共享磁盘的容错功能可以通过共享磁盘子系统上的 RAID 得到实施。

以下示例可以说明 NeoKylinHA 提供的一些好处。

假设你已经配置了有 3 台服务器的集群，在集群中每个服务器上都安装了 Web 服务。在集群中的每个服务器上拥有两个网站服务，它包括所有数据，图形和网页内容，让每个网站都连接到存储到集群中每个服务器上的共享磁盘子系统。下图 1-1 将描绘出这个构造组成：

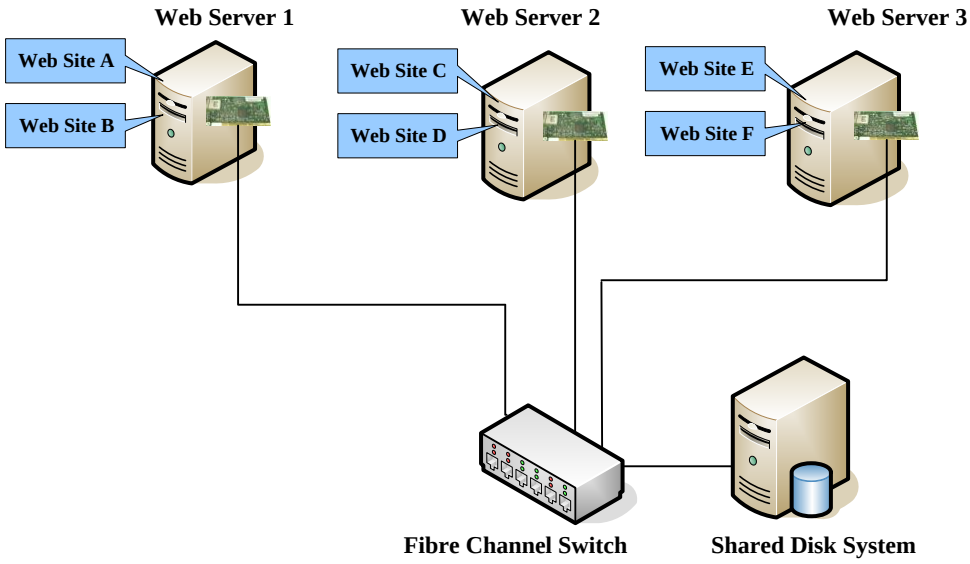

图 1-1 集群配置示例

在正常的集群运转中，每个集群中的服务器是在不断的彼此沟通，并定期执行检测所有已知资源，以便于发现故障者。

假设 Web 服务器 1 发生硬件或软件的问题致使通过 Web 服务器 1 的用户与 Internet，电子邮件和网上信息失去了联系。下图 1-2 将显示说明当 Web 服务器 1 失效时，系统资源将

如何迁移：

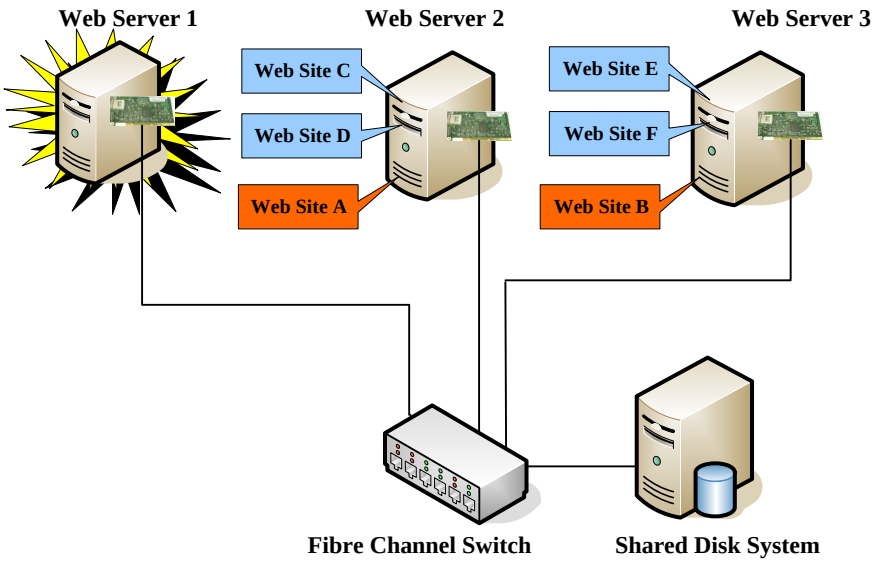

图 1-2 集群服务器失效时资源迁移示例

网站 A 迁移到 Web 服务器 2 上、网站 B 迁移到 Web 服务器 3 上，相应的 IP 地址和证书也迁移到 Web 服务器 2 和 Web 服务器 3 上。

当您配置集群时，你需要确定哪个 Web 服务器上的网站将发生故障。在前面的示例中您配置的网站 A 迁移到 Web 服务器 2 和网站 B 迁移到 Web 服务器 3。这样，通过 Web 服务器 1 的处理的工作将平均分配给集群中的其它成员并在其上继续运行。

1.3 集群配置

NeoKylinHA 集群中可能包括也可能不会包括一个共享的磁盘子系统。共享磁盘子系统可以通过光纤传导或使用 iSCSI 来进行连接。如果集群中的一台服务器失效，另一个集群中被指定的服务器将自动挂载失效服务器原在共享磁盘所含的目录内容。这样就使网络用户可以的不间断地进入共享磁盘子系统的该目录内容。典型的 NeoKylinHA 资源可能包括数据，应用程序和服务。下图 1-3 将显示一个典型的光纤集群的组成：

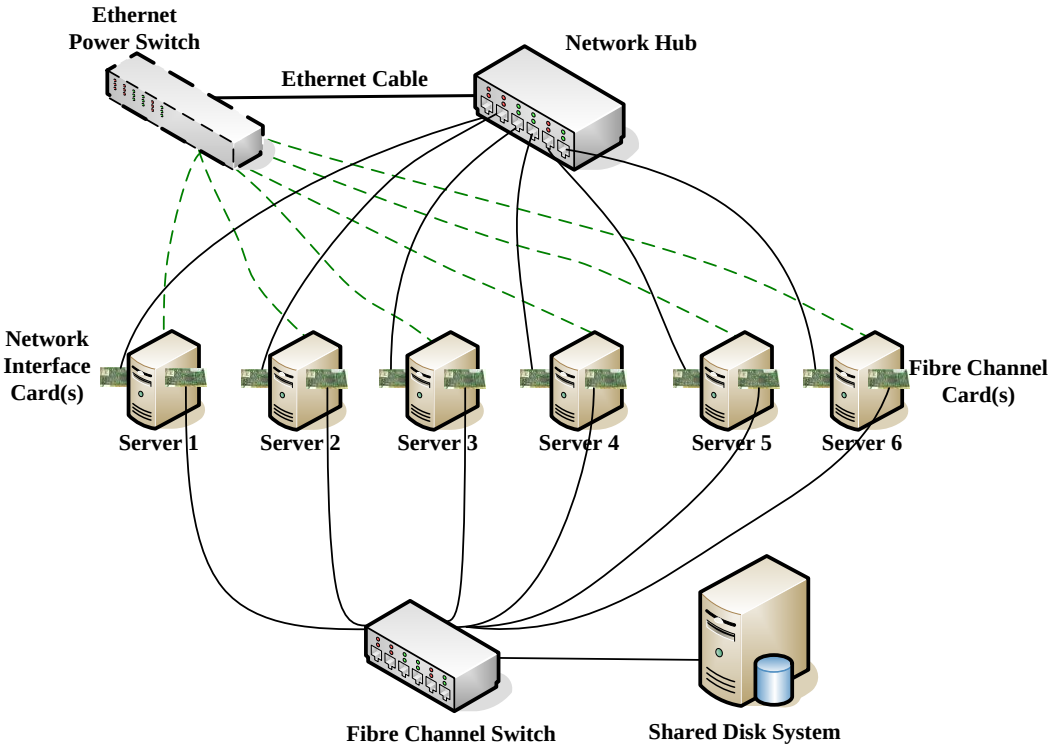

图 1-3 典型的光纤集群组成示例

除此之外我们还可以用 iSCSI（即 internet Small computer systems interface，是 IETF 制订的一项标准，用于将 SCSI 数据块映射成以太网数据包）来实现集群，虽然性能不如光纤，但成本更低。下图 1-4 将显示这种构造：

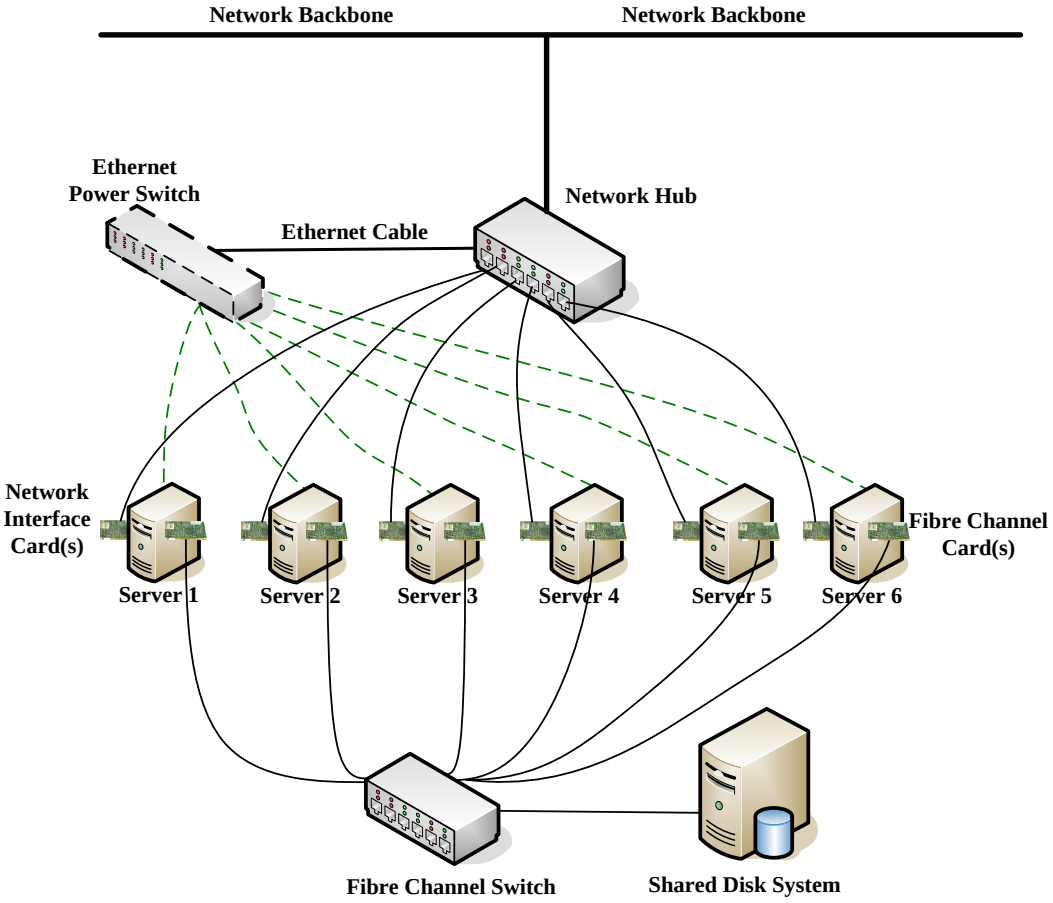

图 1-4 iSCSI 构造集群示例

虽然大多数的集群，包括一个共享的磁盘子系统，但也可以创造一个没有共享的磁盘子系统的 NeoKylinHA 集群。下图 1-5 展示一个没有共享的磁盘子系统的 NeoKylinHA 集群：

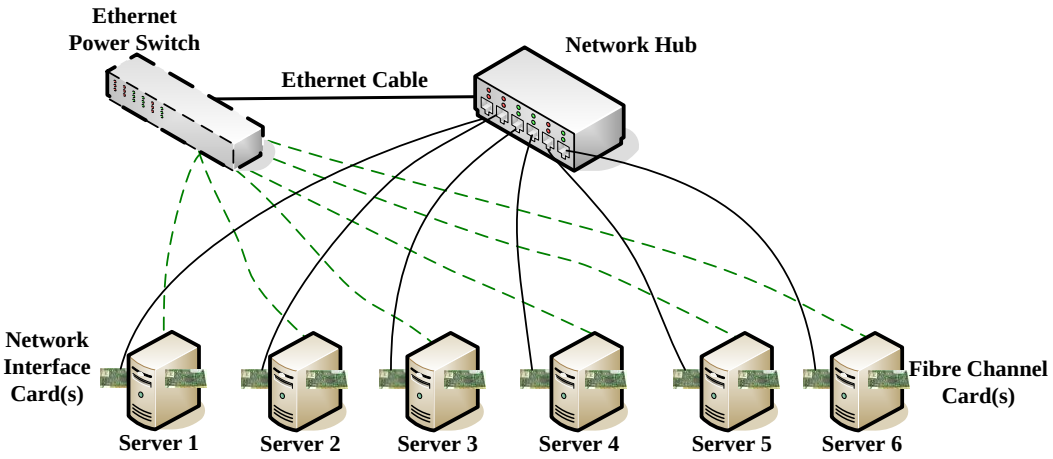

图 1-5 没有共享的磁盘子系统的集群示例

1.4 高可用集群组成

下面是搭建 NeoKylinHA 集群环境的条件：

⑴ 一个集群中需包含 2-- 16 台服务器；

⑵ 集群中的服务器均需安装 NeoKylinHA 集群软件；

⑶ （可选）共享磁盘；

⑷ （可选）光纤存储；

⑸ 连接到集群中的服务器需要保证至少两种以上的方式可以使 NeoKylinHA 的服务器连通，这包括：以太网，串行连接等。

1.5 体系结构

本节提供 NeoKylinHA 体系结构简介，将介绍 NeoKylinHA 组件的基本信息和描述这些组件是如何互通的。

1.5.1 架构层次

NeoKylinHA 有一个分层架构。图 1-6 说明不同层次及其相关的组件。

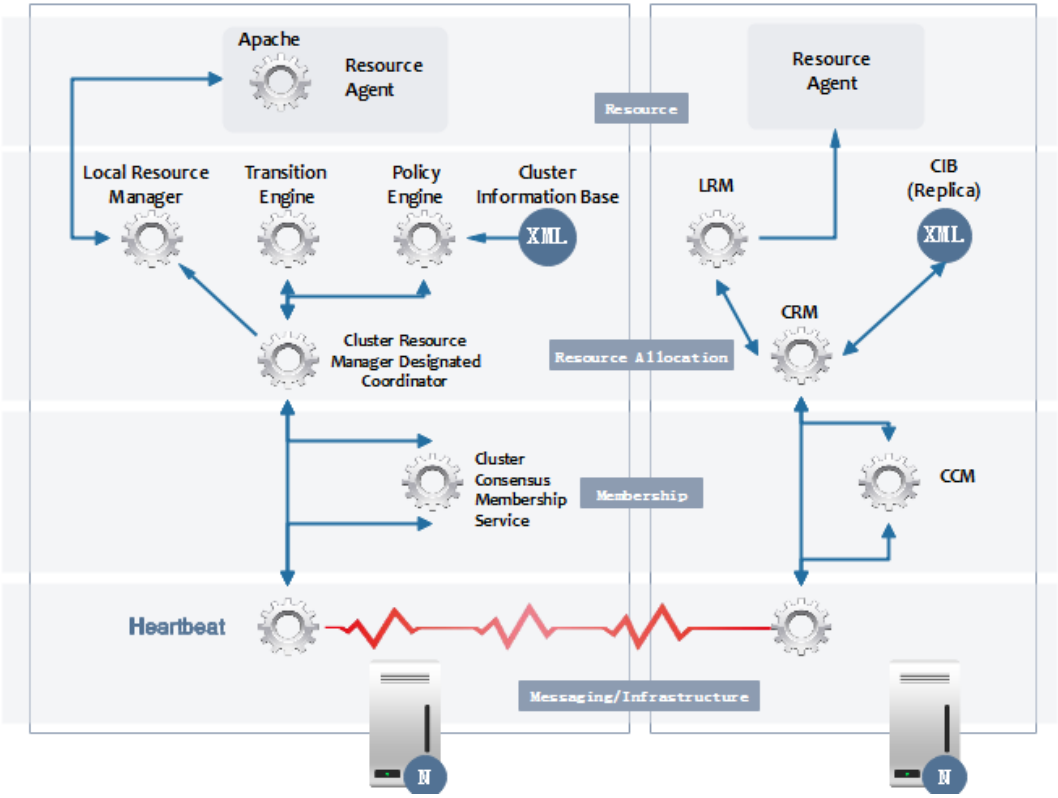

图 1-6 NeoKylinHA 底层架构

通讯/基础设施层

图中底下第一层是通讯/基础设施层，也称为作为心跳层。这一层包含的组件发送心跳信息及其它资料，来向集群发布自己的工作状态。心跳检测就处于该层。

成员层

图中底下第二层是成员层。成员层负责计算最大的充分连通集群节点和同步观察其所有成员。在基于从心跳层获得的资料来执行这一任务。逻辑上照顾集群服务，并为集群更高层提供一个有组织的集群拓扑结构（节点方式）。

资源分配层

第三层是资源分配层。这层是整个集群构成中最复杂的，它由下列组件构成：

集群资源管理（CRM）

资源分配层的每一个动作都要通过集群资源管理。如果资源分配层其它组成部分或更高层的组件资源需要沟通，它们必须通过集群资源管理来进行。

对每个节点，集群资源管理是保持集群信息的基础，即 CRM。一个集群资源管理在集群中被选为主控节点（DC），也就是说它具有 CRM。群集中的其它 CRM 是主 CRM 的副本。

所有对 CRM 的修改必须对主 CRM 文件进行修改。DC 是集群中唯一具有决定权的实体，譬如隔离某个节点，转移资源等。

集群信息基准

集群信息基准(CIB)是一个 XML 数据文件，用于存储集权配置、状态、节点、资源、限制条件等。集群中只有一个主 CIB 文件，是通过 DC 来维护的。所有节点的 CIB 文件都是主 CIB 文件的副本。如果管理员需要修改集权配置，就必须利用 DC 来对主 CIB 文件进行修改，可以利用命令行方式或者图形界面工具。

策略引擎（ PE ）和转换引擎（TE）

当主控节点（DC）需要对集群做出更改时（作用于新的 CIB），策略引擎来计算下一状态中群集和列表（资源）需要采取的行动，以便于通过这一目的来命令计算机做出变化，然后执行转换引擎。DC 发送消息给有关联的 CRM，然后 CRM 调用本地资源管理（LRM）完成对资源的修改。PE 与 TE 双方达到最合理的运行在 DC 节点上。

本地的资源管理（LRM）

本地资源管理调用本地资源代理为 CRM。它可以在 CRM 上执行启动/停止/监察运转和报告结果。LRM 包含这本地节点的状态信息。

资源层

第四层是资源层也是最高的层。资源层包括一个或更多的资源代理（RA）的。资源代理是一种程序，通常是一个 shell 脚本， 可以通过脚本来启动，停止和监测某种服务（资源）。

1.5.2 工艺流程

在集群中的许多行为的执行将导致集群性质的变化。这些行动可以包括添加、删除集群资源或修改资源。在集群众当您执行这些行操作是，重要的是要了解发生什么。

举例来说，假设您要添加群集 IP 地址资源。为此，您使用 cibadmin 命令行工具或图形工具来更改集群主 CIB 文件。但不须使用 cibadmin 命令或使用图形工具对 DC 更改。您可以使用集群中任何节点上的工具的，将主 CIB 所要求的变更传往指定 DC，然后复制到所有集群节点，并开始切换程序。

然后，其它节点的 CRM 利用自身的 LRM 更改 CIB 文件并将报告返回至 DC 以通知结果。当 DC 中的 TE 成功执行完毕，集群就会回到空闲状态等待其它事件发生。如果没有安装计划那样进行执行，PE 将会重新启动。当一个应用死掉或着节点宕机，DC 会通过 CCM 得到节点宕机的通知，通过 LRM 得到应用死掉的通知，然后 DC 将决定如何做切换，进入

到下一个集群状态。新的集群状态通过新的 CIB 文件来描述。

2 系统环境搭建说明

本章将介绍部署 NeoKylinHA 高可用集群有关的硬件和软件等方面的要求。

2.1 硬件要求

下面指定的硬件要求表示的是高可用服务器最低硬件配置。在此基础上可以根据自己的要求对集群进行相应的硬件扩充。

⑴至少有两个 Linux 服务器。服务器不需要必须是相同的配置（内存，磁盘空间等）。

⑵至少有两个传输媒介（以太网等），通过它们使集群节点互相连通。传输媒介应支持数据率在 100Mb/s 或更高。

⑶一个 STONITH（shoot the other node in the head）装置。

2.2 软件要求

确保以下软件要求得到满足：

中标麒麟 Linux 服务器操作系统（NeoShine Linux Server）；

中标麒麟高可用软件产品（NeoKylinHA）。

2.3 共享磁盘要求

如果您想要的数据拥有高度可用性建议为你的集群创立共享磁盘。共享的磁盘子系统是用来确保以下内容：

⑴共享磁盘，根据制造商的说明需要事先配置。

⑵磁盘载于共享磁盘系统应配置为使用镜像或 RAID 添加容错，以便共享磁盘系统。建议使用基于硬件的 RAID ，基于软件的 RAID 不支持所有的配置。

⑶如果您正在使用的共享磁盘系统 iSCSI，应确保事先配置正确。

2.4 操作系统网络配置说明

(1) 必须保证每个网卡有一个固定的 IP 地址，该地址用于对外提供相应的服务或心跳；

(2) 除了连接公网的网卡之外，每台计算机至少另外配置一块网卡或者串口作为专用于

集群内部的通信，如果是双机配置，私网之间可以通过交叉网线来连接；

(3) 公网与私网建议使用不同的 TCP/IP 子网地址，如图 2-1 为两节点的网络配置示例。

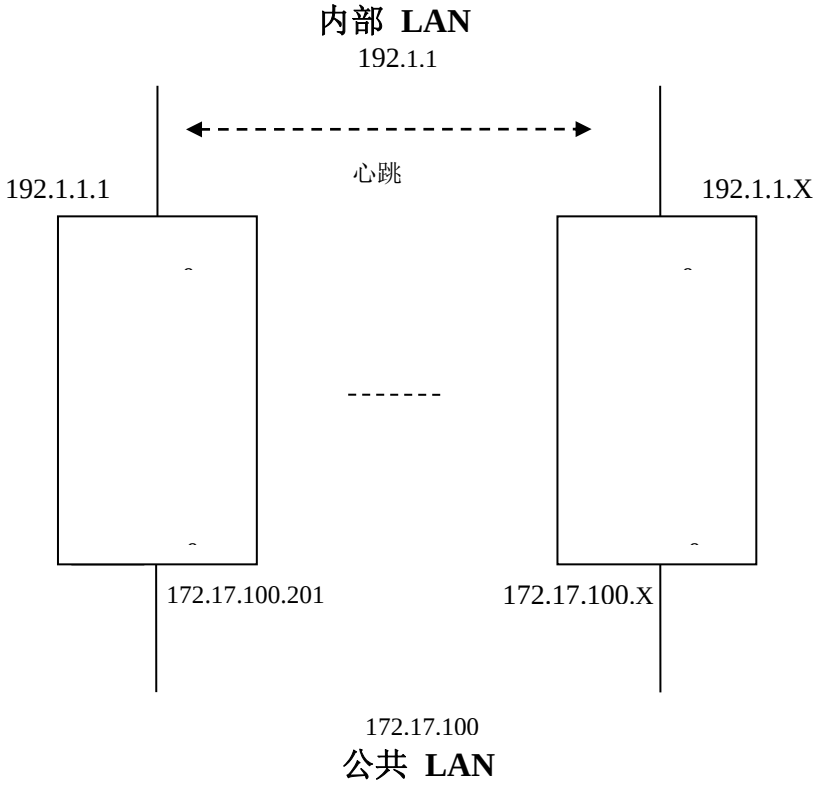

图 2-1 网络配置示例图

3 高可用集群配置和管理

中标麒麟 Linux 高可用服务器配置指南可以帮助管理员迅速有效的创建高可用集群。本指南提供最为常见的配置说明，但不包括所有可能存在的场景。以下内容将以基本的 NFS 文件服务为例，按部就班的讲解如何创建一个低成本的高可用集群。

我们需要以下配置：

1、两台包含冗余硬件的机器，如双网卡等；

2、使用 drbd 作数据镜像；

3、同一时间内只允许一台机器读取数据；

4、提供一单独 IP 地址。

在开始配置之前，请按照第二章和第三章中的内容把两台机器安装配置到位。附加说明一点，每个机器需要提供一相同的分区以备配置 drbd 之需。

配置说明分两个主要部分：配置高可用资源和创建限制条件。

所有的配置将会以 XML 文件方式存储。为方便起见，依赖关系要单独的加载到集群配置中。

3.1 基础配置

集群管理工具被分割为两部分，配置与管理。管理部分包含资源状态以及资源属性等，管理工具可构建实时的集群状态。状态显示部分通过集群中每个节点的本地资源管理进程（lrmd）提供。集群有可能重新载入整个部分，所以这些不会写入磁盘并告诫管理员禁止修改。

配置部分包含基本的集群控制选项，资源列表，资源属性以及限制条件等。此部分主要的焦点问题被分为以下 4 部分内容：

1、基本配置选项；

2、节点；

3、资源；

4、限制条件（可以理解为资源之间的关系）。

以下为配置文件基本构架（空配置）示例：

```
<cib generated="true" admin_epoch="0" epoch="0" num_updates="0"
have_quorum="false">
<configuration>
<crm_config/>
<nodes/>
<resources/>
<constraints/>
</configuration>
<status/>
</cib>
```

3.1.1 集群实时状态

在配置集群之前，需要解释如何用命令行查看状态。使用 crm_mon 工具可以显示正在运行的集群状态，包括资源状态，节点状态等信息，另外可利用它检测集群中不正常的状态。详细参数执行 crm_mon --help 即可。

3.1.2 更新配置

下面是一则警告：

警告：更行配置规则

请不要手动编辑 cib.xml 文件，否则集群会拒绝使用该配置文件。如有需要，使用 cibadmin 进行配置。

利用 cibadmin 命令可对正在运行中的集群进行配置。cibadmin 提供执行效率高的查询，添加，删除，升级以及替换功能。简单的使用 cibadmin 分以下三步走：

1、保存当前配置文件

```
[root@server1 ~]# cibadmin --cib_query > /tmp/tmp.xml
```

2、编辑保存的 xml 文件

利用 xml 工具或其它文本编辑工具进行编辑（可以利用 xml 编辑工具的 DTD 功能进行

语法标识，dtd 文件存放于/opt/NeoKylinHA/lib/heartbeat/crm.dtd）

3、更新配置

```
[root@server1 ~]# cibadmin --cib_replace --xml-file /tmp/tmp.xml
```

如果只需要编辑资源配置部分，可利用下面方式：

```
[root@server1 ~]# cibadmin --cib_query --obj_type resources >
/tmp/tmp.xml
[root@server1 ~]# vi /tmp/tmp.xml
[root@server1 ~]# cibadmin --cib_replace --obj_type resources
--xml-file /tmp/tmp.xml
```

3.1.3 迅速删除某段配置

有些时候需要快速删除一部分配置，可以进行如下 3 步：

1、标识出那些兑现是要删除的：

```
[root@server1 ~]# cibadmin -Q | grep stonith
<nvpair id="cib-bootstrap-options-stonith-action" name="stonith-action"
value="reboot"/>
<nvpair id="cib-bootstrap-options-stonith-enabled" name="stonith-enabled"
value="1"/>
<primitive id="child_DoFencing" class="stonith" type="external/vmware">
<lrm_resource id="child_DoFencing:0" type="external/vmware" class="stonith">
<lrm_resource id="child_DoFencing:0" type="external/vmware" class="stonith">
<lrm_resource id="child_DoFencing:1" type="external/vmware" class="stonith">
<lrm_resource id="child_DoFencing:0" type="external/vmware" class="stonith">
```

```
<lrm_resource id="child_DoFencing:2" type="external/vmware" class="stonith">
<lrm_resource id="child_DoFencing:0" type="external/vmware" class="stonith">
<lrm_resource id="child_DoFencing:3" type="external/vmware" class="stonith">
```

2、标识出资源标记和 id

本例子中为 primitive 和 child_DoFencing

3、执行 cibadmin

```
[root@server1 ~]# cibadmin --cib_delete --crm_xml '<primitive
id="child_DoFencing"/>'
```

3.1.4 不利用 XML 更行配置

利用其它高级工具同样可以更行配置文件，从而避免编辑复杂的 XML 文件。例如：

启动 STONITH：

```
[root@server1 ~]# crm_attribute --attr-name stonith-enabled
--attr-value true
```

查看某节点是否处于备机状态：

```
[root@server1 ~]# crm_standby --get-value --node-uname somenode
```

查询某资源的位置限制条件：

```
[root@server1 ~]# crm_resource --locate --resource my-test-rsc
```

3.1.5 测试配置

在更改完配置以后，验证其正确性是很有必要的。下面将讲解如何验证：

1、保存当前配置：

```
[root@server1 ~]# cibadmin --cib_query > /tmp/tmp.xml
```

2、利用 XML 编辑器编辑该文件。

3、利用 ptest 工具进行验证：

```
[root@server1 ~]# ptest -VVVVV --xml-file /tmp/tmp.xml
--save-graph tmp.graph --save-dotfile tmp.dot
```

该工具使用与集群相同底层库文件来模拟利用此配置文件所发生的情况。tmp.graph 与 tmp.dot 是所输入的两个重要文件。两个文件用来描述根据配置文件集群所做出的响应。graph 文件是存储转换完成情况，包含一系列的动作，参数以及先决条件。虽然 graph 文件不易读懂，但可以利用 dot 文件来理解相同的意思。

3.2 命令行配置管理资源

中标麒麟高可用集群软件软件提供 3 中不同的资源代理脚本。一是早期的中标麒麟高可用集群软件软件遗留脚本，二是 Linux 的 LSB 脚本，三是开放集群框架脚本，简称 OCF。一下内容将以 LSB 和 OCF 脚本为主，来讲解如何配置高可用资源。

3.2.1 LSB 资源代理脚本

所有的 LSB 脚本都可在 Linux 系统的/etc/init.d 下找到。LSB 脚本必须包含 start，stop，restart，reload，force-reload，status 执行部分。

/etc/init.d 下的服务并不一定符合标准情况，所以在利用 LSB 脚本的时候，管理员要充分理解和掌握技术要领。利用中标麒麟高可用集群软件软件进行配置/etc/init.d 下的服务可能出现错误，所以，需要手动确认需要配置的服务是否是系统自动启动的。如果是被系统在开始时候启动的，需要手动配置为停止状态以及不随系统启动而启动，这样才能避免出现不必要的麻烦，切记。另外，在配置 LSB 资源之前，需确认是否集群中每个节点都存在该 LSB 脚本（中标麒麟高可用集群软件软件不负责此部分）。

3.2.2 OCF 资源代理脚本

所有的 OCF 资源代理脚本存放在/opt/NeoKylinHA/lib/ocf/resource.d/heartbeat 下。OCF 脚本类似于 LSB 脚本，但是 OCF 可以输入参数。OCF 资源脚本至少包含 start，stop，status，monitor 以及 meta-data 执行动作。其中，meta-data 动作给出如何配置该脚本，例如：

```
[root@server1 ~]# /opt/NeoKylinHA/lib/ocf/resource.d/heartbeat/IPaddr meta-data
```

上述命令输出为 XML 格式。可利用 ra-api-1.dtd 来验证 XML 的正确性。它包含 3 部分：首先是描述部分，然后是所有的参数，最后为动作部分。

一个典型的 OCF 资源代理脚本的 meta-data 输出为：

```
<!DOCTYPE resource-agent SYSTEM "ra-api-1.dtd">
<resource-agent name="apache"> ❶
<!-- Some elements omitted -->
<parameter name="ip" unique="1" required="1">❷
<longdesc lang="en">❸
The IPv4 address to be configured in dotted quad notation, for example
"192.168.1.1".
</longdesc>
<shortdesc lang="en">IPv4 address</shortdesc>
<content type="string" default="" />❹
</parameter>
</resource-agent>
```

这是 IPaddr 代理脚本的一部分。下面详细解释一下：

❶ 根元素；

❷ 配置参数为 ip，这是该资源代理脚本的强制部分；

❸ 参数描述；

❹ 参数类型以及缺省值；

在第 3 章中可找到配置资源代理实例。

3.2.3 NFS 服务配置实例

搭建 NFS 服务器，需要文件系统资源，drbd 资源和 NFS 服务 IP 地址组成的组资源。管理原可以针对每个资源进行单独的配置，然后通过 cibadmin 进行导入，命令行如下：

```
[root@server1 ~]# cibadmin -C -o resources -x resource_configuration_file
```

3.2.4 配置文件系统资源

文件系统资源是一个 OCF 资源代理脚本，通过它可以实现挂载卸载文件系统。在本实例中磁盘设备为/dev/drbd0，挂载点为/srv/failover，文件系统为 reiserfs。

资源配置如下：

```
<primitive id="filesystem_resource" class="ocf" provider="heartbeat"
type="Filesystem">
<instance_attributes id="ia-filesystem_1">
<attributes>
<nvpair id="filesystem-nv-1" name="device" value="/dev/drbd0"/>
<nvpair id="filesystem-nv-2" name="directory" value="/srv/failover"/>
<nvpair id="filesystem-nv-3" name="fstype" value="reiserfs"/>
</attributes>
</instance_attributes>
</primitive>
```

3.2.5 NFS 服务和 IP 地址

配置 NFS 服务两台机器需要使用相同的 IP 地址，并且他们必须同时运行在一个机器上。在这种状况下，启动次序并不是非常重要的，他们可以同时被启动，所以可以将 NFS 服务

IP 地址放入一个组中来管理。

在开始配置之前，需要确保 NFS 能够正常提供服务，详细请参考相关 NFS 手册即可。NFS 服务是 LSB 资源代理脚本，IP 地址为 OCF 资源代理脚本。

```
<group id="nfs_group">❶
<primitive id="nfs_resource" class="lsb" type="nfsserver"/>❷
<primitive id="ip_resource" class="ocf" provider="heartbeat"
type="IPaddr">❸
<instance_attributes id="ia-ipaddr_1">
<attributes>
<nvpair id="ipaddr-nv-1" name="ip" value="10.10.0.1"/>❹
</attributes>
</instance_attributes>
</primitive>
</group>
```

❶ 组资源标识；

❷ nfsserver 资源基本元素；

❸ IPaddr OCF 资源代理脚本不需要任何配置，只需配置 instance_attributes；

❹ 这是 IPaddr 中唯一一处配置参数。更多的参数可以在资源代理脚本的 metadata 中找到。

3.2.6 配置 CRM 选项

CRM 选项是整个集群的全局变量。领论上缺省的参数可用于多数的集群环境，但是如果需要提供特殊服务必须更改些选项值，比如 STONITH 等。所有的选项可以通过 crm_config 工作以 nvpair 为标识添加到 cib.xml 中。例如，更改 cluster-delay 从 60 秒到 120 秒，用如下配置：

```
<cluster_property_set>
<attributes>
<nvpair id="1" name="cluster-delay" value="120s"/>
</attributes>
</cluster_property_set>
```

将此内容写入一文本文件，然后利用 cibadmin -C -o crm_config -x filename 进行导入。

下面详细介绍所有 crm 参数：

cluster-delay（间隔时间，缺省 60s）：该参数与 transiton_idle_timeout 一样需要众所周知。如果在这断时间内没有任何响应，通讯即被认为出现问题。如果任何初始化操作需要明显的高延迟的话，该值要调高。

symmetric_cluster（布尔值，缺省为 TRUE）：如果为真，资源可缺省运行与任何节点上。否则，需要明确设置限制条件，资源才能被启动。

stonith_enabled（布尔值，缺省为 TRUE）：如果为真，将隔离失效的节点。

no_quorum_policy（枚举类型，缺省为 stop）：有如下三个取值。

a. ignore，继续认定仲裁有效。

b. freeze，不启动任何资源，但可以迁移资源。Fencing 被禁止。

c. stop，停止所有正在运行的资源。Fencing 被禁止。

default_resource_stickiness（整型，缺省等于 0）：资源粘贴值，用于描述资源是否可以从已运行的节点迁移到其它“更好“的节点。主要分以下几种情况：

a. 0，资源运行在哪个节点上有系统来决定。

b. 大于 0 的值，资源选择继续运行在当前节点上的权重，越高的值，当前节点就具有更高的选择权。

c. 小于 0 的值，资源选择不继续运行在当前节点上的权重，越高的绝对值表明当前节点有更高的遗弃权。

d. INFINITY，正无穷，资源会一直运行在当前节点上，除节点宕机，节点转换为备机或者配置改变。

e. -INFINITY，资源总是从当前节点迁移走。

is_managed_default（布尔值，缺省为 TRUE）：

a. TRUE，资源可启动，关闭，监控，迁移等。

b. FALSE，资源不受集群管理，如当停止的时候不会再启动等。

stop_orphan_resources（布尔值，缺省为 TRUE），如果某个资源被发现没有被定义任何选项，则：

a. TRUE，停止该资源。

b. FALSE，不理睬该资源。

此参数多用于当某个资源被删除而不需要先执行停止该资源的操作。

stop_orphan_actions（布尔值，缺省为 TRUE），如果发现当前执行的动作为被定义，则：

a. TRUE，停止该动作。

b. FALSE，不理睬改动作。

3.2.7 限制条件的配置

当配置完资源以后，需要配置限制条件，这样集群才会明确资源的运行规则。没有限制条件的配置，资源会没有规则的运行在节点上，势必可能会发生意想不到的错误。

在中标麒麟高可用集群软件中存在 3 中不同的限制条件：

1、位置限制条件，定义资源运行的节点；

2、协同限制条件，定义哪两个资源是否运行于同一个节点；

3、顺序限制条件，定义资源的启动顺序。

3.2.7.1 位置限制条件

该限制条件可针对每个资源添加多次。所有的的位置限制条件需要对给予的资源赋值。简单的实例，资源 filesystem_1 在节点 earth 上的分值为 100：

```
<rsc_location id="filesystem_1_location" rsc="filesystem_1">❶
<rule id="pref_filesystem_1" score="100">❷
<expression attribute="#uname" operation="eq" value="earth"/>❸
</rule>
</rsc_location>
```

❶ 位置限制条件的基本标识；

❷ 设置分值为 100，分值计算请参见“分值计算规则”；

❸ 设置表达式属性。

3.2.7.2 协同限制条件

协同是用于定义资源是否可以运行在同一个节点上的，它只能定义 INFINITY 或者 -INFINITY 值，来表明运行在同一个节点和不能运行在同一个节点。实例如下，资源 filesyste_resource 和 nfs_group 总是运行在同一个节点上：

```
<rsc_colocation
id="nfs_on_filesystem"
to="filesystem_resource"
from="nfs_group"
score="INFINITY"/>
```

针对主从模式的资源配置需要事先明确当前节点是否为主模式方式运行，可通过附加的 to_role 或者 from_role 属性来判定。

3.2.7.3 顺序限制条件

有些时候有些资源需要其它资源启动以后才能启动，这样就需要利用顺序限制条件来解决此问题。顺序限制条件可以不同的次序启动或者关闭不同的资源，可以是启动或者关闭或者提升为主模式等。例如：

```
<rsc_order id="nfs_after_filesystem" from="group_nfs" action="start"
to="filesystem_resource" to_action="start" type="after"/>
```

利用 type="after"，可以指定 from 的资源在 to 的资源之后启动。

3.2.7.4 限制条件配置实例

本章的 drbd 实例如果没有添加限制条件是无法启用的。它必须指定所有的资源必须运行在 drbd 主模式所在的节点上。或这说，在其它资源启动之前，该节点上的 drbd 服务必须启动为主模式才可。因为只有 drbd 主模式上的机器上才能挂载磁盘。下面是详细限制条件设置：

- 文件系统资源必须与主模式的 drbd 资源运行于同一个节点。

```
<rsc_colocation id="filesystem_on_master" to="drbd_resource"
to_role="master" from="filesystem_resource" score="INFINITY"/>
```

- 文件系统必须在 drbd 资源被提升为主模式以后才能被挂载。

```
<rsc_order id="drbd_first" from="filesystem_resource" action="
to="drbd_resource" to_action="promote" type="after"/>
```

- NFS 服务于 IP 地址必须在文件系统资源启动以后启动。

```
<rsc_order id="nfs_second" from="nfs_group" action="start"
to="filesystem_resource" to_action="start" type="after"/>
```

- NFS 服务和 IP 地址要在同一个机器上运行。

```
<rsc_colocation id="nfs_on_drbd" to="filesystem_resource"
from="nfs_group" score="INFINITY"/>
```

- 可附加一条，用于防止 NFS 服务运行于从模式的 drbd 节点。

```
<rsc_colocation id="nfs_on_slave" to="drbd_resource"
to_role="slave" from="nfs_group" score="-INFINITY"/>
```

4 高可用集群管理工具

本章主要介绍在 CIB 和集群资源管理配置中的必要工具，其它用于管理资源应用和调试排错的工具在第八章增加资源和第七章常见问题解答中有详细说明。

以下介绍几个集群管理相关的任务，并简要的介绍如何使用工具来完成这些任务。

集群状态监控

crm_mon 命令主要用于集群状态监控和配置。该命令输出包括节点数目、uname、uuid、状态、集群中的资源配置和目前的状态。crm_mon 的输出能够在字符终端下显示或者记录在 HTML 文件中。当节点状态在配置文件中没有被定义，crm_mon 在文件的指定区域增加节点和资源的描述。这个命令的用法和语法详细介绍参考 crm_mon(8)。

管理 CIB

cibadmin 命令是操作 CIB 的基础管理命令。它能够用来转存、更新、修改全部或部分 CIB 内容，删除整个 CIB 内容，或者执行各种 CIB 的管理操作。这个命令的用法和语法详细介绍参考 cibadmin（8）。

管理配置转化

crm_diff 是帮助生成和应用 XML 补丁的命令。这是用于显示集群配置两个版本之间的不同变化或者使用 cibadmin（8）命令保存最后一次应用配置。这个命令的用法和语法详细介绍参考 crm_diff（8）

修改 CIB 属性

crm_attribute 命令可以对 CIB 进行查询和修改。管理员利用 crm_attribute 修改 CIB 中的“nodes”、“status”、“crm_config”段。详细见后。

验证集群配置

crm_verify 命令用于验证 CIB 的一致性、检测其它错误和测试是否可以联机到正在运行的集群。它反馈两类问题。只有将错有错误全部解决才能保证集群的运行。管理员可以将正

在使用的 CIB 复制出来进行修改，然后利用 crm_verify 对新更改的 CIB，确保无误以后可以利用 cibadmin 将新 CIB 文件导入到正在运行的集群中。详细见后说明。

管理资源配置

crm_resource 负责与 CRM 进行交互。可以启动、停止、删除或者迁移资源在集群节点中。详细见后说明。

管理资源错误统计

crm_failcount 命令可以查询当前资源错误统计。详细见后说明。

生成节点 UUIDs

UUIDs 是项的唯一标识。crm_uuid 用于显示节点的 UUID。详细见后说明。

管理节点备机状态

crm_standby 命令用来控制备机属性。无论如何资源可以被运行在某个节点上取决于该节点在 CIB 中备机属性。资源只能运行在非处于备份状态。当一个节点处于备份状态，资源不能被运行，但是可以执行其它的操作比如内核升级等。移除备份状态即可实现主机状态。详细见后说明。

4.1 cibadmin－读取

cibadmin－读取，修改和管理 NeoKylinHA 的 CIB。

详解

cibadmin 是对 CIB 进行修改维护的主要命令。它对 CIB 的 XML 树进行操作，简化手动编写 XML 文件的工作。

--id xml-object-id,-i xml-object-id，指定 XML ID，下个版本中会去除。

--obj_type object-type,-o object-type，指定对象属性，属性值为“节点”“资源”“状态”“限制条件”。

--verbose,-V，显示调试信息。

--help,-？，显示命令帮助信息。

--cib_erase,-E，擦除整个 CIB。

--cib_query,-Q，查询 CIB。

--cib_create,-C，创建新的 CIB 文件。

--cib_replace,-R，递归替换 CIB 中的对象。

--cib_update,-U，递归更新 CIB 中的对象。

--cib_modify,-M，修改 CIB 中对象属性，利用—id 选项指定。

--cib_delete,-D，删除指定的第一个匹配对象。

--cib_delete_alt,-d，删除全部匹配对象，需要-o 参数。

--cib_ismaster,-m，显示是否为主 CIB 实例，返回 0 为主实例。

--cib_sync,-S，强制同步 CIB 文件（-h 指定节点，无-h 为利用 DC）。

--crm_xm xml-fragment-string,-X xml-fragment-string，指定一个 XML 段落。

--xml-file filename,-x filename，指定 XML 文件。

--xml_pipe,-p，指定冲管道方式获取该 XML 文件。

其它专用选项

--cib_bump,-B，强制增加 CIB 计数。

--cib_master,-w，强制 CIB 可读可写（危险操作）。

--cib_slave，-r，强制 CIB 实例只读（危险操作）。

--force_quorum,-f，强制写入 CIB，不管集群是否存在仲裁（需谨慎使用，否则将要产生冲突）。

--host hostname,-h hostname，指定主机。

--local,-l，本地执行。

--no-bcast,-b，不对其它主机进行广播通知。

--sync-cal,-s，在退出之前等待传输给 cibadmin 的指令。

4.2 crmadmin－控制 CRM

详解

--help,-?，显示帮助信息。

--verbose,-V，显示调试信息。

--quiet,-q，屏蔽调试信息。

--bash-export,-B，导入到 Bash 环境变量，为 crmadmin -N 所用。

--debug_inc node,-i node，增加 CRM 守护进程调试等级。

--debug_dec node,-d node，减少 CRM 守护进程调试等级。

--kill node,-K node，关掉指定节点的 CRM。

--status node,-S node，查询指定节点的 CRM 状态。

--election node,-E node，打开指定节点的选举功能。

--dc_lookup,-D，查询当前 DC 的信息。

--nodes,-N，查询所有节点的信息。

4.3 crm_attribute－手动维护 CIB

命令可以对 CIB 进行查询和修改。管理员利用 crm_attribute 修改 CIB 中的“nodes”、“status”、“crm_config”段。

详解

--help,-?，显示帮助信息。

--verbose,-V，显示调试信息。

--quiet,-Q，利用-G 参数进行查询的时候，只输出值。

--get-value,-G，检索信息。

--delete-attr,-D，删除属性。

--attr-value string,-v sting，设置属性，于-G 冲突。

--node-uuid node_uuid,-u node_uuid，指定节点的 UUID。

--node-uname node_uname,-U node_uname，指定节点的 uname。

--set-name string,-s string，指定设置属性。

--attr-name string,-n string，指定设置或查询的属性。

--type string,-t type，限定 CIB 那些数据段可以被设置，值包括 nodes,status 或者 crm_config。

4.4 crm_diff－区别集群配置文件与接受配置文件补丁

crm_diff 用以帮助集群管理员合并不同的配置至 CIB 中。它可以比较两个配置文件的不同之处，可以创建补丁。

详解

--help,-?，显示帮助信息。

--original filename,-o filename，指定原有文件。

--new filename,-n filename，指定新文件名。

--original-string string,-O string，指定原有字段。

--new-string string,-N string，指定新字段。

--patch filename,-p filename，对原有 XML 打补丁，与-o 合用。

--cib,-c，作为 CIB 输入。

--stdin,-s，从标准输入中读取输入。

4.5 crm_failcount－在指定的资源上进行错误统计

NeoKylinHA 利用复杂的方法指定错误切换到那个节点上。当资源设置了资源粘性以后，根据数值大小判断优先级。带有资源错粘性属性，根据值来决定触发切换。错误统计是资源监视器的附加属性，它与资源错误粘性数值相乘得到结果为错误切换分值。如果这个数值超过设置的大小，资源就会发上切换，除非错误统计数被重置。crm_failcount 负责查询给定节点上的每一个资源的错误记录。要将资源会切，需删除错误统计数。

详解

--help,-?，显示帮助信息。

--verbose,-V，显示调试信息。

--quiet,-Q，利用-G 进行查询时，只显示数值。

--get-value,-G，查询检索。

--delete-attr,-D，删除属性。

--attr-value string,-v sting，指定属性字段，与-G 冲突。

--node-uuid node_uuid,-u node_uuid，指定 UUID。

--node-uname node_uname,-U node_uname，指定 uname。

--resource-id resource name,-r resource name，指定资源名称。

4.6 crm_master－指定那个资源可以被加载

通过调用外部资源代理程序来决定那个资源可以被执行。它不建议冲命令行执行，只是作为一个资源代理程序的帮助工作而已。通过设置有效时间周期，可以指定在重启以后继续运行（生存周期设置为“永远”）或者重启以后不自动运行（生存周期设置为“重启”）。

详解

--help,-?，显示帮助信息。

--verbose,-V，显示调试信息。

--quiet,-Q，利用-G 参数进行查询的时候，只输出值。

--get-value,-G，检索信息。

--delete-attr,-D，删除属性。

--attr-value string,-v sting，设置属性，于-G 冲突。

--lifetime string,-l string，指定时间周期，有 reboot 与 forever 两个值。

4.7 crm_mon－集群状态监视器

利用 crm_mon 命令可以配置和监视集群状态。该命令输出节点数量、名称、UUID 以及状态。crm_mon 的所收集的数据能够很容易提供给显示程序。crm_mon 输入的信息可以在终端显示或者输出为 HTML 页面。通过 XML 集群配置文件可以创建一个关于节点和资源的预览。

详解

--help,-?，显示帮助。

--verbose,-V，显示调试信息。

--interval seconds,-i seconds，是监视时间间隔，缺省为 15 秒。

--group-by-node,-n，资源分组显示。

--inactive,-r，显示不活动的资源。

--as-console,-c，在终端上显示集群状态。

--one-shot,-1，显示一次状态即退出。

--as-html filename,-h filename，写入指定的 html 文件。

--daemonize,-d，后台执行。

--pid-file filename,-p filename，指定集群配置 XML 文件。

4.8 crm_resource

crm_resource－与 CRM 进行交互，列出资源，启动，停止或迁移资源。

详解

--help,-?，显示帮助信息。

--verbose,-V，显示调试信息。

--quiet,-Q，只输出值，与-W 同用。

--list,-L，列出所有资源。

--query-xml,-x，查询一个资源，需要-r 参数同时执行。

--locate,-V，定位一个资源，需要-r 参数同时执行。

--migrate,-M，迁移一个资源，利用-H 指定目标节点。如果-H 没有被指定，则强制资源进行转移通过当前位置和-INFINITY 分值创建的规则。

需要-r 参数，可选参数-H，-f

--un-migrate,-U，移除所有通过-M 设置的限制，需要-r,-t 参数。

--delete,-D，重 CIB 中删除一个资源，需要-r,-t 参数。

--cleanup,-C，重 LRM 中删除一个资源，需要-r 参数，可选参数-H。

--reprobe,-P，从 CRM 外部重新检测已启动的资源，可选参数-H。

--refresh,-R，从 LRM 中刷新 CIB，可选参数-H。

--set-parameter string,-p string，设置资源参数，需要-r,-v，可选-i,-s。

--get-parameter string,-g string，获得资源参数，需要-r，可选-i,-s。

--delete-parameter string,-d string，删除资源参数，需要-r,可选-i。

--get-property sting,-G sting，获取被命名的属性。

--set-property sting,-S string，设置被命名的属性。

--resource sting,-r string，指定资源 ID。

--resource-type sting,-t sting，指定资源类型（原始的，副本，组等）。

--property-value string,-v string，指定属性值。

--host-uname string,-H string，指定主机名。

--force-relocation,-f，强制资源进行转移，通过当前位置和-INFINITY 分值所创建的规则，用于当资源粘性与限制分数超过 INFINITY(100,000)的时候。

-s string，指定实例属性的 ID。

-i string，指定 nvpair 对象的属性。

4.9 crm_standby

crm_standby－对节点的备机属性进行错作并决定资源是否可以运行在该节点上。无论如何资源可以被运行在某个节点上取决于该节点在 CIB 中备机属性。资源只能运行在非处于备份状态。当一个节点处于备份状态，资源不能被运行，但是可以执行其它的操作比如内核升级等。移除备份状态即可实现主机状态。通过设置有效时间周期，可以指定在重启以后继续运行（生存周期设置为“永远”）或者重启以后不自动运行（生存周期设置为“重启”）。

详解

--help,-?，显示帮助信息。

--verbose,-V，显示调试信息。

--quiet,-Q，利用-G 参数进行查询的时候，只输出值。

--get-value,-G，检索信息。

--delete-attr,-D，删除属性。

--attr-value string,-v sting，设置属性，于-G 冲突。

--node-uuid node_uuid,-u node_uuid，指定节点的 UUID。

--node-uname node_uname,-U node_uname，指定节点的 uname。

--lifetime string,-l string，指定时间周期，有 reboot 与 foreve 两个值。

4.10 crm_uuid－获取节点 UUID

详解

直接执行即可。

4.11 crm_verify－验证 CIB 一致性

用于验证 CIB 的一致性、检测其它错误和测试是否可以联机到正在运行的集群。它反馈两类问题。只有将错有错误全部解决才能保证集群的运行。管理员可以将正在使用的 CIB 复制出来进行修改，然后利用 crm_verify 对新更改的 CIB，确保无误以后可以利用 cibadmin 将新 CIB 文件导入到正在运行的集群中。

详解

--help,-?，显示帮助信息。

--verbose,-V，显示调试信息。

--live-check,-L，测试集群连接。

--crm_xml string,-X string，根据提供的字段检测配置，只能通过完整的 CIB。

--xml-file file,-x file，检测配置文件。

--xml-pipe,-p，通过标准输入使用配置，只能通过完整的 CIB。

5 资源保护脚本简介

在高可用集群中对应用的保护和管理都是通过创建和管理对应的资源来实现的，在高可用集群中提供了资源代理（resource agents）和 STONITH 代理（STONITH agents），你可以通过编写自己的资源代理和 STONITH 代理脚本来扩展高可用集群对不同应用保护的支持。

注释：什么是 STONITH？

Stonith 即 shoot the other node in the head，在高可用集群中实现节点隔离和阻断的功能。Stonith 设备是一种能够自动关闭电源来响应软件命令的设备。

5.1 STONITH（Fence）代理

当集群检测到集群中的节点出现故障时将删除该节点，我们称之为隔离节点，要实现问题节点的隔离就需要通过使用 STONITH 资源实现该功能。所有 STONITH 资源默认存放在每个节点的/usr/lib/stonith/plugins 目录下。通过这些 STONITH 资源可是实现节点关机、节点重启等功能。

我们可以使用 SSH STONITH 代理进行 STONITH 功能的测试，可以通过 SSH STONITH 代理实现关闭节点的功能。在实际的生产环境中，我们不建议使用 SSH STONITH 代理进行节点的管理，因为使用 SSH STONITH 代理就需要确保所有系统都可以提供 SSH 服务并且能够通过 SSH 进行访问。

如果想知道集群能够提供那些 STONITH 设备的支持，可以使用 stonith –L 命令进行查看。

文档附录二中将有详细描述。

5.2 其它资源代理

集群中所有被保护的资源都是通过资源代理进行管理和监控的，NeoKylinHA 支持两种类型的资源代理：

5.2.1 LSB 脚本

LSB 即 Linux Standard Base (LSB)，LSB 脚本的编写规范和系统启动是运行的服务启动脚本规范相同，系统提供的 LSB 资源脚本存放在/etc/init.d 目录下。

5.2.2 OCF 脚本

在高可用集群中，我们建议用户在进行资源保护时采用 OCF 资源代理脚本实现对资源的管理和监控。OCF（Open Cluster Framework project）脚本编写规范是基于 LSB 的扩展，系统提供的 OCF 资源代理脚本存放在/usr/lib/ocf/resource.d 下。

6 常见问题

在我们实际使用高可用软件过程中可能遇到很多无法理解的问题，中标麒麟高可用软件提供很多有用的工具帮助用户了解高可用软件内容运行的情况。

6.1 高可用集群的运行情况？

要查看当前高可用集群的运行情况，我们可以使用 crm_mon 命令查看。通过该命令可以查看 DC 运行的节点、节点和资源的运行情况等。

6.2 集群的中的一些节点无法看到其它节点？

出现该问题的主要原因可能是节点之间的通信（比如心跳）出现问题，通常情况下是由于节点配置了防火墙导致节点间通信出现问题。当然如果集群中出现脑裂（Split-Brain）的情况也会导致该问题的出现。

注释：什么是脑裂（Split-Brain）？

当节点之间都不能正确获取到对方节点状态时会出现脑裂的情况。可以通过使用 fencing 设备或采用创建多个通信链路的方法解决通信的单点故障导致集群出现脑裂的问题。了解更多信息请登陆

6.3 要列出当前已知资源

要列出当前系统已知资源可以使用 crm_resource －L 命令，执行结果示例如图 6-1 所示。

图 6-1 列出当前已知资源

6.4 配置的一个资源总是提示运行失败？

当添加一个资源时如果提示运行失败，请先尝试通过手动运行资源代理脚本的方法进行测试，以确定脚本的正确性。如果使用 LSB 脚本，通过运行 scriptname start、scriptname stop 起停服务。如果使用 OCF 脚本，请先设置相关的环境变量，然后手动运行脚本进行测试。例如：要测试 IPaddr OCF 脚本，你需要先设置以 OCF_RESKEY_开头的 OCF_RESKEY_ip 变量。请运行以下命令：

```
[root@server1 ~]# export OCF_RESKEY_ip=<your_ip_address>
[root@server1 ~]# /usr/lib/ocf/resource.d/heartbeat/IPaddr validate-all
[root@server1 ~]# /usr/lib/ocf/resource.d/heartbeat/IPaddr start
[root@server1 ~]# /usr/lib/ocf/resource.d/heartbeat/IPaddr stop
```

如果运行失败，很可能你没有按照要求提供所需变量的定义或者设置参数时键入了错误的变量定义内容。

6.5 如果提示错误信息，如何获取更多有效的错误信息？

你可以在运行命令时添加-V 选项，系统将提示更多调试信息。

6.6 如何清理我的资源？

你可以通过 crm_resource －L 获取你的资源的 ID，然后通过 crm_resource －C －r 资源 id 删除该资源。

附录一　术语表

CRM：cluster resource manager

从这个名字就可以看出这个模块是整个 HA 的一个中心，整个系统的一个大脑了，他主要负责整个系统的各种资源的当前配置信息，以及各个资源的调度。也就是根据各资源的配置信息，以及当前的运行状况来决定每一个资源（或者资源组）到底该在哪个节点运行。不过这些事情并不是他直接去做的，而是通过调度其它的一些模块来进行。通过 HA 模块来进行节点之间的通信，调度节点之间的工作协调。随时将通过 heartbeat 模块收集到的各个成员节点的基本信息转交给 CCM 某块来更新整个集群的 membership 信息。他指挥 LRM（local resource manager）对当前节点的各资源执行各种相应的操作（如 start、stop、restart 和 monitor 等等），同时也接收 LRM 在进行各种操作的反馈信息并作出相应的决策再指挥后续工作。另外 CRM 模块还负责将各个模块反馈回来的各种信息通过调用设定的日志记录程序记录到日志文件中。

LRM：local resource manager

LRM 是整个 HA 系统中直接操作所管理的各个资源的一个模块，负责对资源的监控，启动，停止或者重启等操作。这个模块目前支持有四种类型的资源代理（resource agent）：

Heartbeat

ocf（open cluster framework）

lsb（Linux standard base）

stonith

四种类型的 resource agent 中的前三种所调用的脚本分别存如下路径：

heartbeat：/etc/ha.d/resource.d/

ocf：/usr/lib/resource.d/heartbeat/

lsb：/etc/init.d/

LRM 就是通过调用以上路径中存储的各种脚本来实现对资源的各种操作。每一种类型的脚本都可以由用户自定义，只要支持各类型的标准即可。实际上这里的标准就是接受一个标准的调用命令和参数格式，同时返回符合标准的值即可。至于脚本中的各种操作时如何的 LRM 并不关心。

PE：CRM Policy Engine

他主要负责将 CRM 发过来的一些信息按照配置文件中的各种设置来进行计算，分析。然后将结果信息按照某种固定的格式通过 CRM 提交给 TE（Transition Engine）去分析出后续需要采取的相应的动作。PE 需要计算分析的信息主要是当前有哪些节点，各节点的状况，当前管理有哪些资源，各资源当前在哪一个节点，在各个节点的状态如何等等。

TE：Transition engine

主要工作是分析 PE 的计算结果，然后根据配置信息转换成后续所需的相应操作。个人感觉 PE 和 TE 组合成一个类似于规则引擎实现的功能，而且 PE 和 TE 这两个模块只有在处于 active 的节点被启动。另外 PE 和 TE 并不直接通信，而都是通过高可用的指挥中心 CRM 来传达信息的。

CIB：cluster information base

CIB 在系统中负责保存当前集群中各资源原始配置以及动态变化了的状态，统计信息收集分发中心，是一个不断更新的信息库。当他收集到任何资源的变化，以及节点统计信息的变化后，都会集成整合到一起组成当前集群最新的信息，并分发到集群各个节点。分发动作并不是自己和各个节点通信，同样也是通过 心跳模块来做的。

CIB 收集整理并汇总出来的信息是以一个 xml 格式保存起来的。该文件实际上就是 CIB 的信息库文件。在运行过程中，CIB 可能会常读取并修改该文件的内容，以保证信息的更新。

CCM：consensus cluster membership

CCM 的最主要工作就是管理集群中各个节点的成员以及各成员之间的关系。他让集群中各个节点有效的组织称一个整体，保持着稳定的连接。Heartbeat 模块所担当的只是一个通信工具，而 CCM 是通过这个通信工具来将各个成员连接到一起成为一个整体。

LOGD：logging daemon（non-blocking）

一个无阻塞的日志记录程序，主要负责接收 CRM 从各个其它模块所收集的相关信息，然后记录到指定额度日志文件中。当 logd 接收到日志信息后会立刻返回给 CRM 反馈。并不是一定要等到将所有信息记录到文件后再返回，日志信息的记录实际上是一个异步的操作。

STONITH

Stonith 即 shoot the other node in the head，在高可用集群中实现节点隔离和阻断的功能。Stonith 设备是一种能够自动关闭电源来响应软件命令的设备。

Split-Brain

脑裂：当节点之间都不能正确获取到对方节点状态时会出现脑裂的情况。可以通过使用 fencing 设备或采用创建多个通信链路的方法解决通信的单点故障导致集群出现脑裂的问题

iSCSI

iSCSI（互联网小型计算机系统接口）是一种在 Internet 协议网络上，特别是以太网上 进行数据块传输的标准。它是由 Cisco 和 IBM 两家发起的，并且得到了 IP 存储技术拥护者 的大力支持。是一个供硬件设备使用的可以在 IP 协议上层运行的 SCSI 指令集。简单地说，iSCSI 可以实现在 IP 网络上运行 SCSI 协议，使其能够在诸如高速千兆以太网上进行路由选择。

IPC

IPC（Internet Process Connection）是共享“命名管道”的资源，通过 IPC 可以实现进程间通信，所以它是为了让进程间通信而开放的命名管道。应用服务创建的消息队列退出后没有删除，我们要使用 ipcrm 命令来删除不再使用的消息队列，可以使用 ipcs 进行消息队列的查询。

附录二　Fence 技术详细描述

屏蔽在 HA（高可用性）计算机群集中是一个非常重要的概念。群集有时候会检测到某个节点功能不正常，需要将其删除。这就称为屏障，它通常通过 STONITH 资源完成。屏障可以定义为一种使 HA 群集具有已知状态的方法。

群集中的每个资源都附带一个状态，例如：“资源 r1 在节点 1 上启动”。在 HA 群集中，这种状态暗示了“资源 r1 在除节点 1 的所有节点上都是停止的”，因为 HA 群集必须确保每个资源最多只能在一个节点上启动。每个节点都必须报告资源发生的每个更改。这样群集状态就是资源状态和节点状态的集合。

如果某些节点或资源的状态无法确定地建立（不论何种原因），就会采取屏障。即使当群集不了解某些节点上发生的状况时，屏障也可以确保这些节点没有运行任何重要资源。

有两类屏障：资源级别屏障和节点级别屏障。

- 资源级别屏障

通过使用资源级别屏障，群集可以确保节点无法访问一个或多个资源。SAN 就是一个典型的示例，屏障操作更改了 SAN 交换机上的规则从而拒绝节点的访问。

通过使用要保护的资源所依赖的常规资源可以完成资源级别屏障。这种资源当然会拒绝在此节点上启动，所以依赖它的资源将不会在同一节点上运行。

屏障和 STONITH **75**

- 节点级别屏障

节点级别屏障确保节点不会运行任何资源。通常用一种很简单但有点极端的方式来完成此种屏障：只需用电源开关重设置节点。这应该是必要的，因为节点可能根本不会作出响应。

STONITH 设备可分为以下类别：

- 电源分配单元 (PDU)

电源分发单元是管理关键网络、服务器和数据中心设备的电源容量和功能的基本元素。它可以提供对已连接设备的远程负载监视和独立电源出口控制，以实现远程电源循环。

- 不间断电源 (UPS)

通过在公共用电不可用时从独立源供电，不间断电源为已连接设备提供紧急电源。

- 刀片电源控制设备

如果是在刀片组上运行群集，则刀片外壳中的电源控制设备就是提供屏障的唯一候选。当然，此设备必须能够管理单个刀片计算机。

- 无人值守设备

无人值守设备（IBM RSA、HP iLO、Dell DRAC）正变得越来越通用，将来甚至可能成为现成计算机的标准设备。然而，它们相比 UPS 设备有一点不足，因为它们与主机（群集节点）共享一个电源。如果节点没有电源，则设想为控制节点的设备就等于没用。在那种情况下，CRM 屏障节点的尝试将是徒劳，而此情况将一直继续，因为所有其他资源操作将等待屏障/STONITH 操作成功完成。

- 测试设备

测试设备仅用于测试目的。它们通常对硬件更加友好。一旦群集进入生产阶段，它们必须替换为真实的屏障设备。

STONITH 实现

- stonithd

stonithd 是一个守护程序，可以由本地进程或通过网络访问。它接受与屏障操作相应的命令：重设置、关闭电源和打开电源。它还可以检查屏障设备的状态。

stonithd 守护程序在 CRM HA 群集中的每个节点上运行。在 DC 节点上运行的 stonithd 实例从 CRM 接收屏障请求。它会对请求作出响应，其他 stonithd 程序将执行所需的屏障操作。

- STONITH 插件

对于每个受支持的屏障设备，都有一个能够控制此设备的 STONITH 插件。STONITH 插件是屏障设备的界面。所有 STONITH 插件都位于每个节点上的 /usr/lib/stonith/plugins 中。所有 STONITH 插件看上去都与 stonithd 一样，但显著区别在于反映了屏障设备的性质。某些插件支持多个设备。ipmilan（或 external/ipmi)就是一个典型的示例，它实施 IPMI

协议并可以控制任何支持此协议的设备。

特殊 STONITH 设备 （SBD）

- **SBD 基本原理介绍**

  SBD （storage-based death） 是一种基于共享存储设备 fencing 机制，是软件级的 fence，可以使用 iscsi，fcoe，fc 等共享块设备当作 sbd 资源。集群中每台主机上都有一个 sbd 的 daemon 用于监控共享存储，当 sbd 监控不到共享存储设备或者共享存储设备上发现有通过 mail slot 信道传递的 fence 请求时，sbd 会立即 fence 主机。SBD 可以运行在虚拟环境中，虚拟机间或虚拟机与物理机间只要共享存储存在就可以使用。

- **SBD 部署要求**

  1. 必须要求有共享存储设备，可以是 iscsi，fc，fcoe 等
  2. 可以使用一块或者多块 sbd 磁盘，大小约为 1G 左右
  3. SBD 设备不能使用 host-based raid 设备
  4. 集群中不能有 drbd 资源
  5. Sbd 理论上不能超过 255 个集群节点