

IAAS平台基础运维管理手册

1. 文档描述

|  |  |
| --- | --- |
| 标题 |  |
| 项目名称 |  |
| 类别 | 规范文档 □ 安装文档 □ 配置文档 □  设计方案 □ 实施文档 □ 测试文档 □  需求文档 □ 行政管理 □ 其他 ■ |
| 摘要 |  |
| 创建日期 |  |
|  |  |
| 项目执行部门 |  |
| 文档作者 |  |
| 文件名称 |  |

1. 文档发布

|  |  |
| --- | --- |
| 存档名称 |  |
| 版本号 | 1.0 |
| 发布时间 |  |
| 文档状态 |  |

1. 修改记录

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 修改日期 | 作者/修改者 | 修改内容 |
| 1.0 | 20200211 |  | 新建 |
| 1.0.1 | 20200317 |  | 补充SANC架构平台开启与关闭 |
| 1.0.2 | 20200904 |  | 补充用户管理 |
|  |  |  |  |

1. 文档审批

|  |  |  |
| --- | --- | --- |
| 项目职位 | 批准人 | 日期 |
|  |  |  |
|  |  |  |
|  |  |  |

**目录**

[1. 文档说明 7](#_Toc50155513)

[1.1. 适用读者 7](#_Toc50155514)

[1.2. 术语和定义 7](#_Toc50155515)

[1.3. 参考文档 7](#_Toc50155516)

[2. 青云系统概述 8](#_Toc50155517)

[2.1. 青云系统架构 8](#_Toc50155518)

[2.1.1. 青云逻辑架构概述 8](#_Toc50155519)

[2.1.2. 青云部署架构概述 11](#_Toc50155520)

[2.2. 青云软件定义网络 11](#_Toc50155521)

[2.2.1. 虚拟网关 12](#_Toc50155522)

[2.2.2. 青云网络规划 12](#_Toc50155523)

[2.2.3. 外部网络（EIP） 13](#_Toc50155524)

[2.2.4. 物理网络 13](#_Toc50155525)

[2.2.5. 基础网络 13](#_Toc50155526)

[2.2.6. 私有网络 14](#_Toc50155527)

[2.2.7. GRE网络 14](#_Toc50155528)

[2.3. 青云分布式存储 14](#_Toc50155529)

[2.3.1. SDS1.0存储架构 15](#_Toc50155530)

[2.3.2. SDS2.0存储架构 18](#_Toc50155531)

[2.3.3. 存储的安装配置 19](#_Toc50155532)

[2.3.4. 备份与恢复 20](#_Toc50155533)

[2.3.5. 青云数据备份与恢复 21](#_Toc50155534)

[2.4. 青云高可用设计 23](#_Toc50155535)

[2.4.1. 青云物理架构高可用 23](#_Toc50155536)

[2.4.2. 青云软件模块高可用 28](#_Toc50155537)

[3. 青云运维管理流程 32](#_Toc50155538)

[3.1. 青云运维体系组织架构 32](#_Toc50155539)

[3.2. 私有云故障处理流程 33](#_Toc50155540)

[3.3. 私有云变更流程 33](#_Toc50155541)

[4. 平台基础状态检查 34](#_Toc50155542)

[4.1. KS节点状态检查 34](#_Toc50155543)

[4.1.1. VG 34](#_Toc50155544)

[4.1.2. Postgresql 35](#_Toc50155545)

[4.1.3. Webservice 36](#_Toc50155546)

[4.1.4. Pgpool 37](#_Toc50155547)

[4.1.5. Proxy 38](#_Toc50155548)

[4.1.6. Seed 39](#_Toc50155549)

[4.1.7. Snapshot 40](#_Toc50155550)

[4.1.8. Zoocassa 41](#_Toc50155551)

[4.1.9. Vbr 44](#_Toc50155552)

[4.2. Hyper节点状态检查 45](#_Toc50155553)

[4.2.1. 节点功能 45](#_Toc50155554)

[4.2.2. 节点架构 45](#_Toc50155555)

[4.2.3. 状态检查 45](#_Toc50155556)

[5. 平台基础状态维护 46](#_Toc50155557)

[5.1. KS节点关机流程 46](#_Toc50155558)

[5.2. Hyper节点关机流程 47](#_Toc50155559)

[5.3. 平台升级 48](#_Toc50155560)

[5.4. 平台扩容 48](#_Toc50155561)

[5.5. 平台关闭 49](#_Toc50155562)

[5.5.1. 将所有计算节点状态设置为故障（faulty） 49](#_Toc50155563)

[5.5.2. 禁用灾难检测和恢复机制 50](#_Toc50155564)

[5.5.3. 关闭控制台服务 51](#_Toc50155565)

[5.5.4. 关闭计算节点 51](#_Toc50155566)

[5.5.5. 关闭管理节点 52](#_Toc50155567)

[5.6. 平台启动 54](#_Toc50155568)

[5.6.1. 检查节点时钟同步 55](#_Toc50155569)

[5.6.2. 检查所有管理节点的状态 55](#_Toc50155570)

[5.6.3. 检查所有hyper节点的状态，包括 55](#_Toc50155571)

[5.6.4. 将计算节点状态改 55](#_Toc50155572)

[5.6.5. 登陆console进行测试 56](#_Toc50155573)

[5.6.6. 恢复灾难检测和恢复机制 56](#_Toc50155574)

[6. 平台用户管理 57](#_Toc50155575)

[6.1. BOSS用户管理 57](#_Toc50155576)

[6.1.1. BOSS概述 57](#_Toc50155577)

[6.1.2. BOSS角色管理 57](#_Toc50155578)

[6.1.3. BOSS用户管理 57](#_Toc50155579)

[6.2. 控制台用户 58](#_Toc50155580)

[6.2.1. 控制台用户概述 58](#_Toc50155581)

[6.2.2. 用户管理 58](#_Toc50155582)

[7. 平台日常维护 63](#_Toc50155583)

[7.1. 日常维护命令 63](#_Toc50155584)

[7.1.1. 切换zone leader 63](#_Toc50155585)

[7.1.2. 数据库操作工具exec\_sql 63](#_Toc50155586)

[7.1.3. 批量执行操作exec\_nodes.sh 64](#_Toc50155587)

[7.1.4. 配置同步 64](#_Toc50155588)

[7.2. 故障处理 67](#_Toc50155589)

[7.2.1. DRBD故障修复 67](#_Toc50155590)

[7.2.2. ZFS故障修复 75](#_Toc50155591)

[7.3. 青云目录结构 76](#_Toc50155592)

[7.4. 青云系统日志 77](#_Toc50155593)

[7.4.1. Supervisor服务说明 77](#_Toc50155594)

[7.4.2. 青云服务模块日志 77](#_Toc50155595)

[7.4.3. 其他重要日志 78](#_Toc50155596)

[7.5. 日常巡检 78](#_Toc50155597)

[7.5.1. 平台状态巡检 78](#_Toc50155598)

[7.5.2. 资源容量监控 82](#_Toc50155599)

[7.6. 迁移虚拟机 89](#_Toc50155600)

[7.6.1. 通过supervisor迁移 89](#_Toc50155601)

[7.6.2. 使用cli迁移 90](#_Toc50155602)

[7.7. 迁移hyper 91](#_Toc50155603)

[7.8. VG节点迁移 91](#_Toc50155604)

[7.8.1. 单个IP手动迁移 91](#_Toc50155605)

[7.8.2. 使用脚本批量迁移 93](#_Toc50155606)

# 文档说明

## 适用读者

有一定青云环境实施经验的人员。

## 术语和定义

无

## 参考文档

无

# 青云系统概述

## 青云系统架构

### 青云逻辑架构概述



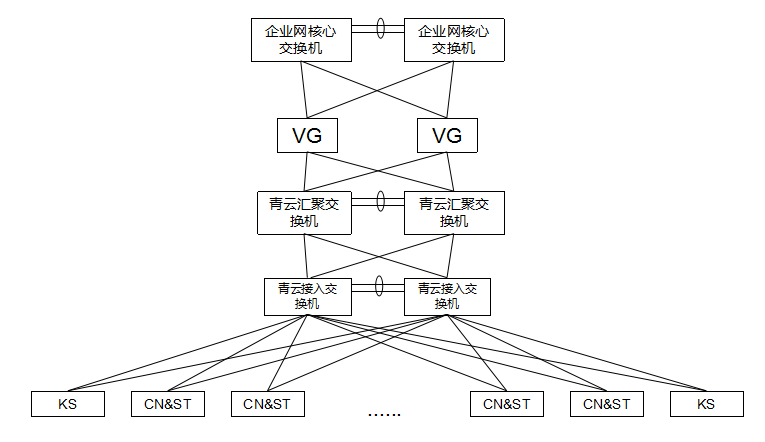
青云平台的架构分为Global和Zone，Global负责青云平台的全局服务，对青云平台外部提供API，并将资源请求转发到不同的Zone，Global中是没有物理资源节点。每个zone有各自的核心服务，为本区域内的资源服务。以下我们简述每个服务的作用：

* 全局服务Global:
  + API server：HTTP API server，对外提供API服务，API server是青云核心系统的接入层，用户的所有请求，包括控制台的请求以及API的请求都会经过API server，再由API server下发到后端服务。
  + WS：global 的消息入口
  + billing：计费模块，此服务主要是通过用户使用的青云系统中的资源来计算所使用的费用，资源主要包括，CPU，内存，磁盘空间，带宽，公网IP等。
  + distributed：分布式锁服务
  + Nf(notify)：通知模块，包括和用户相关的通知和告警
  + boss：boss系统服务模块
  + io：IO和zone里头的PUSH server结合起来完成web push的功能，即资源的动态变化会推送到客户端
  + zk：名字服务，所有qingcloud的服务都会向zk注册自己的信息。
  + pg：关系型数据库，存放用户信息和global的资源信息
  + cass：存放监控数据，允许丢失并最终一致。
  + mc：存放缓存数据，允许丢失
* Zone:
  + fg：zone 的消息入口，zone和global消息交互的唯一通道，主要功能为用户界面和API服务的提供接口。
  + scheduler：实现控制台上“定时器”的功能，该功能可用来定期执行一系列任务，比如定时开关主机、创建备份、调整带宽等。可在指定的周期重复执行，也可仅执行一次，每个任务支持指定多个资源批量处理。执行后会发出通知到用户指定的通知列表并保留近期执行的历史记录。
  + cb：web terminal的proxy server，控制台上“vnc” 的功能，主要功能是使用户可以通过浏览器访问到主机实例。
  + push：配合global的IO server完成web push的功能
  + distributed：分布式锁服务
  + nf：通知模块，和用户无关的通知和告警
  + alarm：告警模块，此服务负责收集报警类信息，包括错误，报警信息。
  + autoscalling：实现控制台上“自动伸缩”的功能，功能可以帮助用户基于资源的监控告警规则动态调节配置或集群规模，比如调整带宽上限，扩容关系型数据库的存储空间，增加或减少负载均衡器后端数量。调节后会发出通知到用户指定的通知列表并保留近期自动伸缩的历史记录。
  + billing\_delegator：实现流量计费的功能
  + zk： 名字服务，所有qingcloud的服务都会向zk注册自己的信 息
  + pg：关系型数据库，存放zone资源信息，例如：虚机/虚拟网络/存储的信息。
  + cass：存放监控数据，数据允许丢失，但是会最终同步一致。
  + mc：存放缓存数据，允许丢失

青云部署在每台服务器的/pitrix目录下，在/pitrix目录下一般会有存在如下子目录，目录说明如下：

|  |  |
| --- | --- |
| 目录 | 目录说明 |
| bin | 放可执行文件和一些辅助脚本 |
| cli | 命令行 |
| conf | 进程配置文件 |
| data/container | 用户数据 |
| data/image | 模板数据 |
| lock | 锁文件 |
| log | 每个进程日志文件 |
| notifier | 告警文件推送目录 |

### 青云部署架构概述



上图为青云典型逻辑部署架构，其中：

* VG:虚拟网关，完成青云内网与青云外网（企业网络）的连接，并部署网络控制器。
* KS:内核服务，青云系统的服务进程。
* CN&ST:计算&存储节点，CN&ST为青云可以对外提供资源的计算与存储节点。

## 青云软件定义网络

青云的内部网络分为三个层次，分别为物理网络、基础网络和私有网络。物理网络是用来连接硬件设备（包括服务器和交换设备）的网络，基础网络是虚拟机／虚拟设备的互联网络，这是一个由青云系统维护的全局网络，私有网络是用户各自自行组建的网络，类似通常所讲的局域网。

物理网络是在安装和配置青云时设置在物理服务器和物理网络设备身上的，其作用是为了物理设备之间的连通。

基础网络无需用户做任何配置与管理，虚拟设备接入其中即可直接使用，但正因为它是全局网络，所以其安全保障是依靠防火墙（Security Group）来做ACL过滤实现的。

私有网络需要用户创建网络设备并进行恰当的管理 （为了简化管理，青云提供了路由器来帮助用户）， 私有网络之间是100%二层隔离（L2 isolation）的，以满足对安全的100%追求。

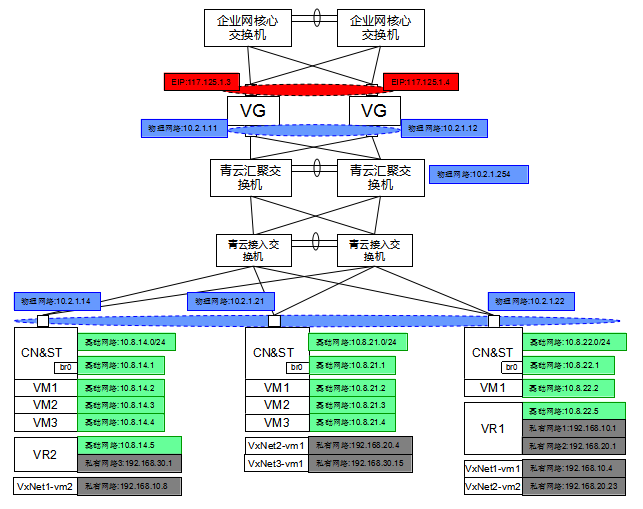
私有网络类似物理世界中使用交换机（L2 Switch）将多台服务器连接在一起， 组成的局域网。它有两种管理方式：受管（Managed）、自管（Unmanaged）。 路由器用于多个受管私有网络之间互联，并提供六项附加服务： DHCP服务、DNS服务、端口转发、VPN、隧道服务、过滤控制，涵盖了常用的网络配置与管理工作。如果青云提供的路由器功能无法满足您对网络管理的需求，您可以创建自管私有网络，并自行配置和管理该网络。

### 虚拟网关

VG位于云内网络和云外网络之间，对进出青云的外网流量进行控制。VG是一组程序，可以部署在任意的服务器上，可以是物理机，也可以是虚拟机。

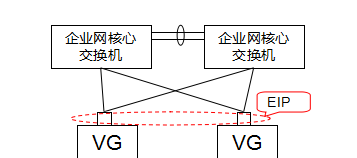
### 青云网络规划

在私有云实施过程中，青云平台的IP地址可以做如下分类：



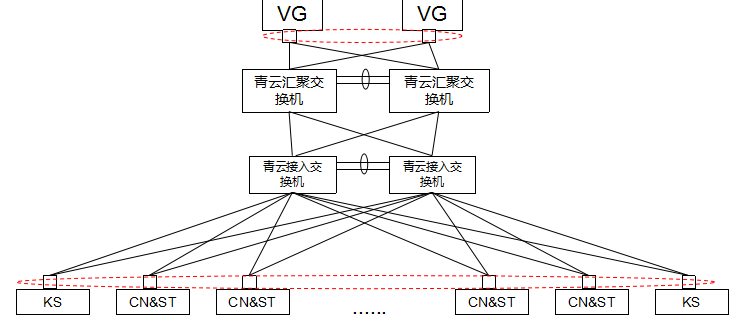
### 外部网络（EIP）

企业内网地址，VG是连接青云平台与企业网，EIP分配给VG上联网卡。从私有云项目实施的角度来看，在VG上联网卡以上的网络为企业内部网（青云外网）。



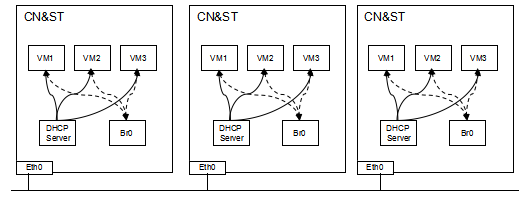
### 物理网络

物理网络地址是青云平台每台宿主机的IP地址。



### 基础网络

这是一个虚拟网络，由青云平台进行维护，分配给每个虚拟机地址。每个物理节点上均启动一个DHCP server，为基础网络中的虚拟机分配地址，同一个物理节点上的虚机通过Bridge互通。在基础网络中，所有基础网络IP均互通。



### 私有网络

这是一个虚拟网络，由用户进行定义私有网络IP地址段，并分配给每个虚拟机。

每台虚拟交换机在部署青云系统的时候，创建在物理服务器上。当用户创建虚拟节点的时候，青云系统会根据系统的资源情况选择物理节点进行创建，并将虚拟机加入虚拟交换机。

用户创建虚拟网络，并将虚拟机加入虚拟网络的时候，每个虚拟机没有私有网络地址。只有将私有网络加入虚拟路由器以后，虚拟路由器上会启动DHCP Server，分配IP地址给私有网络所连接虚拟机。

私有网络在二层是隔离的，私有网络OVS之间通过虚拟路由器互联。

### GRE网络

青云提供隧道服务可以将多个不同地域的局域网连接在一起，形成兼顾公有私有的混合云计算环境，助您有效管理多地域的数据中心。

青云保留10.254.0.0/16地址给GRE隧道做点对点IP使用。在青云网络地址规划的时候，要考虑预留这段地址GRE专用。

## 青云分布式存储

区别于传统的集中式存书，青云系统的存储是分布式存储系统，分布式存储是将数据分散存储在多台独立的物理设备上。

传统的网络存储系统采用集中式的存储服务器存放所有数据，存储服务器成为系统性能的瓶颈，同时集中式存储的造价昂贵，难以满足大规模数据存储应用要求。

分布式网络存储系统采用可扩展的系统架构，利用多台存储服务器分担存储负荷，可以提高系统的可靠性、可用性和存取效率，同时还易于扩展。

青云的分布式存储系统不仅为虚拟主机提供块存储也为对象存储提供存储能力。将全局服务器本地硬盘设备构建成一个统一的资源池，同时分布式存储系统提供数据的多个实时副本，保证用户数据的安全。

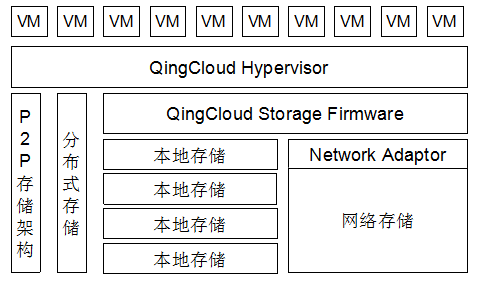
在青云中，硬盘（Volume）为虚拟服务器提供块级的原始存储设，可以被连接到任意运行中的虚拟服务器上，为虚拟服务器提供持久化的，块级存储。

青云提供了3种规格的存储硬盘： 容量型存储、性能存储、超高性能存储。此外针对大数据量的非结构化数据存储提供了专门的对象存储。这几类存储的比较和特点可以参见下图：



### SDS1.0存储架构

青云的环境中使用每一台物理服务器内置的硬盘存储，通过万兆光纤网络连接在一起，再采用自行开发的软件定义存储技术实现了分布式的存储架构，构建了整个云环境的存储资源。青云存储的逻辑架构如下图所示：

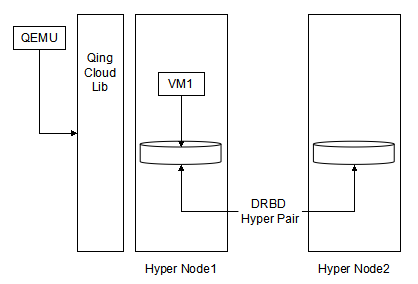


SDS1.0存储架构特点：

* 在物理服务器安装Linux系统时，已经将本地所有的内置硬盘设置为EXT4文件系统。
* 虚拟机采用QCOW2的硬盘格式，以文件形式存放在物理服务器的文件系统上。
* 为了能节约存储空间， 虚拟机采用Sparse file（稀疏文件）的方式。
* 青云提供了一整套存储管理的lib库（Qingcloud Storage Firmware）用于虚拟机存储的管理。 QEMU模块会调用Qingcloud Storage Firmware进行存储管理。
* Qingcloud Storage Firmware会根据事先定义的策略，以及存储监控的实际情况，灵活配置分布式的存储系统。配置时将遵循“距离优先、连续优先、按类别分组”等基本原则。
* 为了保障高可靠性Qingcloud Storage Firmware会为虚拟服务器的磁盘同时维护多个副本，副本的个数默认是1个，可以根据需要进行配置。对于分布式存储的管理采用了Quorum算法， 至少是实时写入一个副本，即只有在主磁盘和副本都写入成功后，这次IO操作才算成功完成。

#### Hyper Pair架构

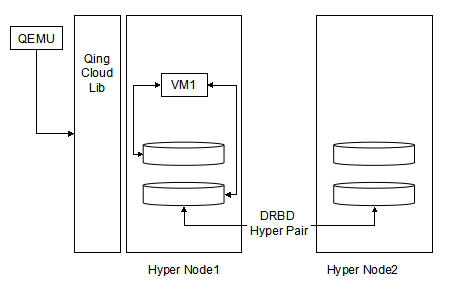
在默认情况下，青云的存储系统会两两配对，既两台服务器组成一个Hyper Pair。虚机在服务器上运行时，硬盘会优先挂在本地，同时在与其配对的服务器上会同步保留一个镜像。



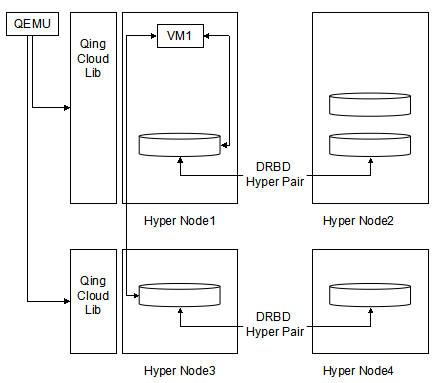
两块硬盘之间的同步通过DRBD服务实现，一个IO写操作，要同时在这两块盘上同时写操作成功，才算最终完成。

#### 新增硬盘时架构

在选择为虚机新增硬盘时，可以选择增加性能硬盘和容量硬盘两种。选择性能硬盘时，考虑到“距离优先”的原则，通常新增的硬盘会与虚机在一台服务器上。



对于增加一块容量型硬盘，通常需要挂接到一台容量型主机上。我们采用深度客户化后的QEMU调用Qingcloud关于存储的底层Lib，实现了分布式的存储架构。



### SDS2.0存储架构

支持资源级副本自定义配置

最大支持同数据中心 4 副本，或者同数据中心 3 副本+异地数据中心 1 副本

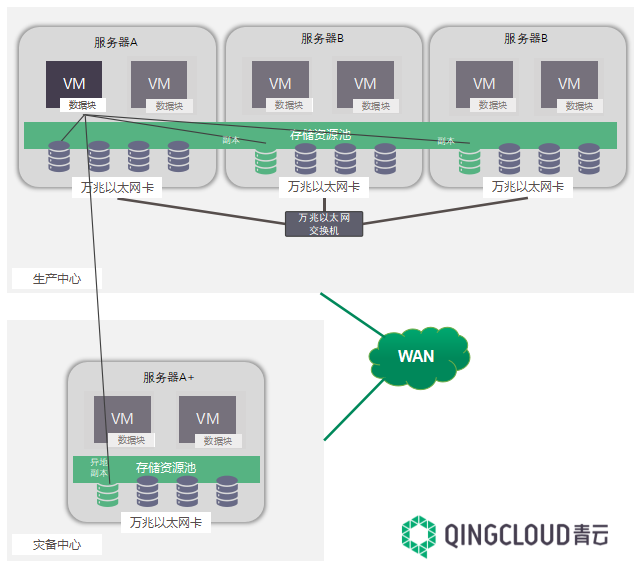
同步时间自定义配置，最小颗粒度 1 秒，同步效率更高

支持压缩， 最大压缩 80%， 平均压缩 50%

热迁移切换时间更短，3 秒内

同时支持容器主机和全虚拟化主机

灾难恢复影响时间更短，30 秒内



### 存储的安装配置

#### SDS1.0

在配置存储时需要注意以下事项:

* 在BIOS中关闭各个磁盘的Disk Cach（可以保证在意外掉电时不丢失数据）
* 打开RAID卡上的NV Cache（ RAID卡上会配置了掉电保护机制）
* 建议物理服务器采用2块盘作为系统盘，这两块盘做成RAID 1的模式。 其余的作为云环境的存储盘，全部作为RAID50模式（至少6块磁盘）。

青云环境中，存储资源的状况是这样的：

* 物理磁盘采用了RAID50的方式， 因此总的容量 = （磁盘总数-1）\* 磁盘容量
* 在青云的存储环境中，可以选择多份镜像副本。 假如选择一份镜像副本， 则有效的存储空间是总存储容量的1/2。

#### SDS2.0

在配置存储时需要注意以下事项:

* 数据盘必须为SSD或flash卡
* 建议物理服务器采用2块盘作为系统盘，这两块盘做成RAID 1的模式。

青云环境中，存储资源的状况是这样的：

* 物理磁盘为采用了直通的方式， 因此总的容量 = 磁盘容量
* 在青云的存储环境中，可以选择多份镜像副本。 假如选择一份镜像副本， 则有效的存储空间是总存储容量的1/2。

### 备份与恢复

备份（Snapshot）用于在块设备级别（block device level）上进行硬盘的备份与恢复， 可以同时对多张硬盘做备份（包括系统盘和数据盘），也可以对正在运行的主机做在线备份。 一张硬盘可以有多个备份链，每条备份链包括一个全量备份点以及多个增量备份点， 您可以随时从任意一个备份点恢复数据。

#### 备份与实时副本的区别

青云的硬盘（包括系统盘和数据盘）都是有多份实时副本的，即数据的写入只有在多个副本上都完成后才算成功，实时副本可以保证硬件设备出现问题时数据不丢失。

备份是用于捕捉硬盘在某一个时刻的状态，未来可以随时恢复到这个状态。在某些时候，例如误操作或者应用逻辑的 bug，可能会导致业务数据的丢失，这种情况下实时副本无法恢复数据，因为硬件设备并没有问题。这时候，就需要通过备份，从历史备份点恢复数据。即，实时副本主要用于灾难恢复；而备份则是用于保存业务数据的里程碑 （milestone）状态，以便在从业务到硬件的任何一个层面出问题时恢复数据。

#### 创建备份

我们可以对一块或多块硬盘（包括系统盘和数据盘）创建备份，我们的备份功能有以下几点特性：

* 基本备份：你可以对主机的系统盘进行备份，也可以对数据盘进行备份。
* 在线备份：你可以对正在运行的主机进行备份，也可以对正在使用的数据盘进行备份。备份操作不会影响或者中断你业务的运行。
* 并行备份：你可以同时对多块硬盘（包括系统盘和数据盘）进行备份，如果这些硬盘同属于一个主机，则在备份时会保证这些硬盘数据的一致性，即都是同一时刻的状态。
* 全量与增量备份：你可以对硬盘（包括系统盘和数据盘）进行全量或者增量备份。当进行全量备份时，会生成一条新的备份链，之后创建的备份默认都是增量备份。

### 青云数据备份与恢复

#### 实时多副本数据

青云系统提供了一整套存储管理的lib库（Qingcloud Storage Firmware）用于虚拟机存储的管理。为了保障高可靠性Qingcloud Storage Firmware会为虚拟服务器的磁盘同时维护多个副本，副本的个数默认是1个，可以根据需要进行配置。对于分布式存储的管理采用了Quorum算法， 至少是实时写入一个副本，即只有在主磁盘和副本都写入成功后，这次IO操作才算成功完成，实时副本可以保证硬件设备出现问题时数据不丢失。

#### 数据备份与恢复

备份是用于捕捉硬盘在某一个时刻的状态，未来可以随时恢复到这个状态。 在某些时候，例如误操作或者应用逻辑的 bug，可能会导致业务数据的丢失， 这种情况下实时副本无法恢复数据，因为硬件设备并没有问题。这时候，就需要通过备份，从历史备份点恢复数据。即，实时副本主要用于灾难恢复；而备份则是用于保存业务数据的里程碑 （milestone）状态，以便在从业务到硬件的任何一个层面出问题时恢复数据。

青云系统提供对一块或多块硬盘（包括系统盘和数据盘）创建备份的功能，青云的备份功能有以下几点特性：

* 基本备份：对主机的系统盘进行备份，也可以对数据盘进行备份。
* 在线备份：对正在运行的主机进行备份，也可以对正在使用的数据盘进行备份。备份操作不会影响或者中断你业务的运行。
* 并行备份：同时对多块硬盘（包括系统盘和数据盘）进行备份，如果这些硬盘同属于一个主机，则在备份时会保证这些硬盘数据的一致性，即都是同一时刻的状态。
* 全量与增量备份：对硬盘（包括系统盘和数据盘）进行全量或者增量备份。当进行全量备份时，会生成一条新的备份链，之后创建的备份默认都是增量备份。



## 青云高可用设计

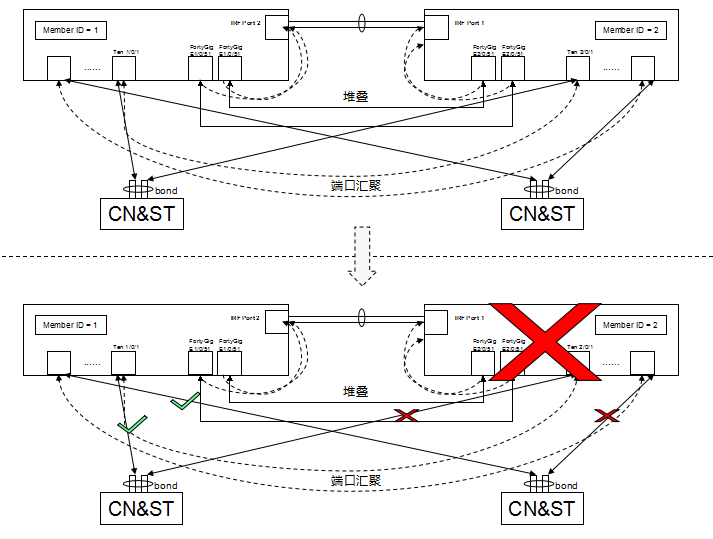
### 青云物理架构高可用

#### 交换机高可用配置

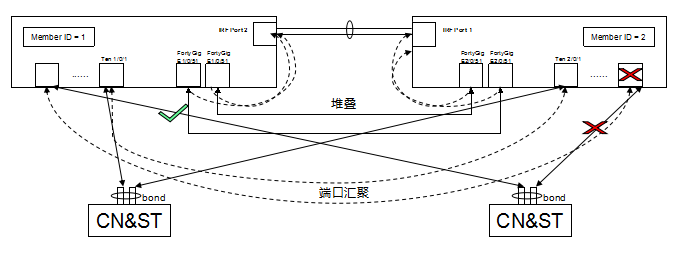
青云交换机配置的高可用是通过交换机堆叠和端口汇聚实现的，交换机堆叠实现交换机级别的高可用，端口汇聚实现端口级别的高可用。

在青云系统中，对青云网络汇聚层/接入层交换机进行堆叠，堆叠技术通过堆叠端口和堆叠连接线将多台交换机连接在一起，通过堆叠选取协议就构成了堆叠，堆叠可以进行集中管理，多台交换机管理起来仿佛一台交换机。

同时更重要的是，交换机堆叠，即虚拟成一个逻辑的交换机，交换机为双活状态（active-active），实现交换机级别的高可用。如果不配置堆叠，而是配置成active-standby，当一个交换机故障，在链路切换过程中会出现闪断。如下图所示，当一个交换机故障，交换机所有端口均无法工作的时候，下联服务器会将流量发送至另外一台堆叠交换机。



同时通过对堆叠交换机进行端口汇聚配置，把以青云交换机的多个属性相同的端口绑定，像一个端口一样使用。端口汇聚提供链路备份和负载分担的功能。配置端口汇聚以后，如果一台交换机的端口故障，与该端口进行端口汇聚的另外一个端口 ，接管所有流量，实现端口级别的高可用。如下图所示，当一个交换机某一端口故障，服务器会将所有流量发送至另外一台堆叠交换机的汇聚端口。

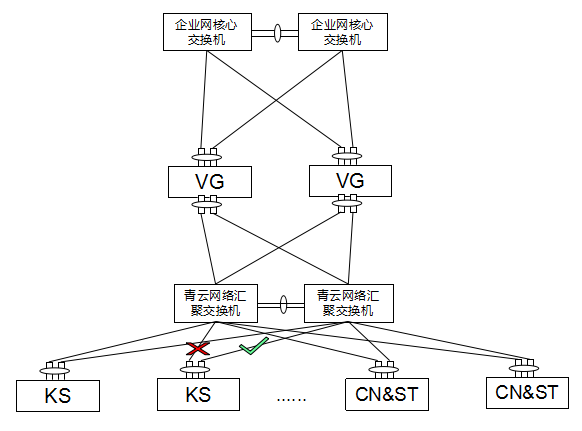


#### 服务器网卡bonding

在青云系统中，VG服务器我们建议配置4块万兆网卡，KS以及CN&ST节点配置2块万兆网卡。并对网卡做bonding，多网卡配置bonding后，被绑定的多个网卡作为一个逻辑网口，对外提供一个统一的网络地址，会大幅提升服务器的网络吞吐(I/O)。同时，更重要的是也提高网络的可用性，例如：当其中一块网卡或者上下联端口故障，不影响网络链路，提高青云系统的可用性。

通常VG上配置4块万兆网卡，两两网卡做bonding，其中的eth0和eth1配置为bond0与青云外（企业）核心交换机对接，eth2和eth3配置为bond1与青云内汇聚交换机对接。交换机需要配置堆叠和端口汇聚，对于企业内部的核心交换机，我们通常只是建议，无法决定是否配置堆叠和端口汇聚。对于青云内部交换机，请配置堆叠和端口汇聚。

在KS和CN&ST节点上通常部署2块万兆网卡，网卡之间做bonding,分别上联青云接入层交换机，两台交换机之间需要配置堆叠，同时不同交换机的对应端口之间做端口汇聚。



#### 服务器硬盘raid

青云系统建议物理节点进行硬盘raid配置，实现数据冗余和备份。在青云系统中，每个物理节点会部署青云软件，同时在CN&ST节点上，通过青云软件创建虚拟资源。我们可以把物理服务器硬盘分类操作系统硬盘（包括青云软件）和存储空间硬盘（创建虚拟资源）。

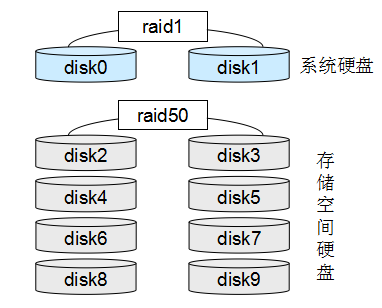
对于部署操作系统的硬盘，做raid1配置，最大限度的保证用户数据的可用性和可修复性，RAID 1的操作方式是把用户写入硬盘的数据百分之百地自动复制到另外一个硬盘上。由于对存储的数据进行百分之百的备份，在所有RAID级别中，RAID 1提供最高的数据安全保障。同样，由于数据的百分之百备份，备份数据占了总存储空间的一半，因而，raid1的磁盘空间利用率低，存储成本高。

对于存储空间硬盘，建议做raid50配置，raid50是RAID5与RAID0的结合。

RAID 5不对存储的数据进行备份，而是把数据和相对应的奇偶校验信息存储到组成RAID5的各个磁盘上，并且奇偶校验信息和相对应的数据分别存储于不同的磁盘上。当RAID5的一个磁盘数据发生损坏后，利用剩下的数据和相应的奇偶校验信息去恢复被损坏的数据。RAID 5的磁盘空间利用率要比RAID 1高，存储成本相对较低，但是raid5两个磁盘数据发生损坏后，数据将丢失。

RAID 0把连续的数据分散到多个磁盘上存取，系统有数据请求就可以被多个磁盘并行的执行，每个磁盘执行属于它自己的那部分数据请求。这种数据上的并行操作可以充分利用总线的带宽，显著提高磁盘整体存取性能，RAID 0提供最高的存储性能。

通过raid50的配置既考虑了数据的备份和安全，有最大限度的提高了数据的存储性能。



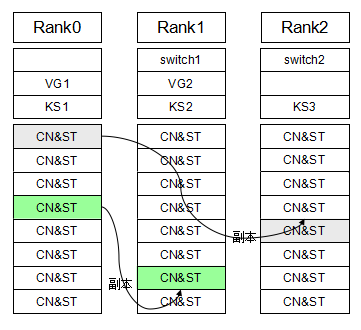
同时在磁盘配置的时候，需要注意以下两点：

* 在BIOS中关闭各个磁盘的Disk Cache，从而保证在意外掉电时不丢失数据；
* 打开RAID卡上的NV Cache(非易失性缓存)，RAID卡上配置掉电保护机制，NV Cache缓存可以在断电情况下继续保存其内部的数据。

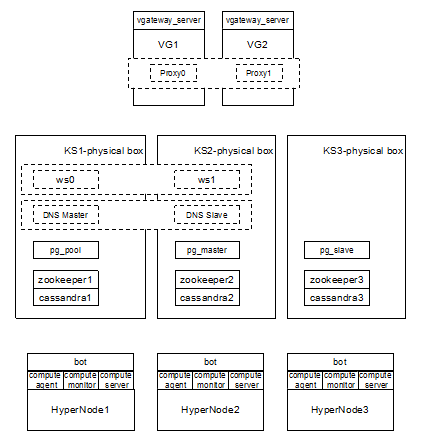
#### 服务器部署

青云的物理服务器部署设计将服务器按照角色的不同分布在不同机柜中，以达到跨机柜容灾的目的，建议汇聚交换机放在排头机柜。

青云物理服务器的角色可以分为VG、KS和CN&ST节点。物理服务器的部署原则，建议VG物理服务器不部署在同一机柜内，KS不部署在同一机柜内。



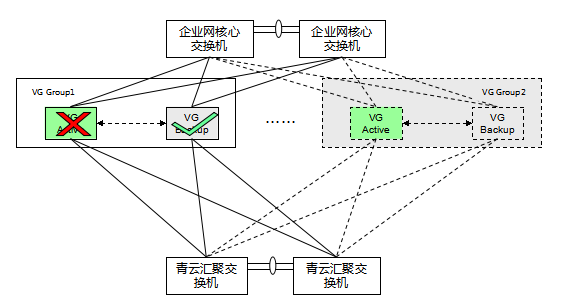
下图是物理服务器部署青云软件模块示意图。



### 青云软件模块高可用

#### VG热备设计

青云VG完成青云内网与青云外网（企业网络）的连接，并部署网络控制器。VG组中的两台VG为一主一备设计，切换时间缺省定义为37秒，其中30秒用于检查确认主节点工作状态（为了避免因为网络抖动导致误切换，30秒为较为安全的时间），7秒为切换时间。

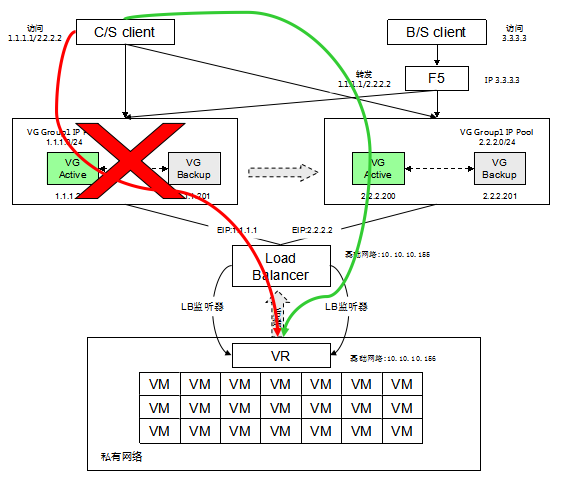


#### VG高可用设计

在基础架构层设计中，可以灵活设计部署两组VG（每组中两台互为主备），每组VG分别负责一个IP段的流量转发。

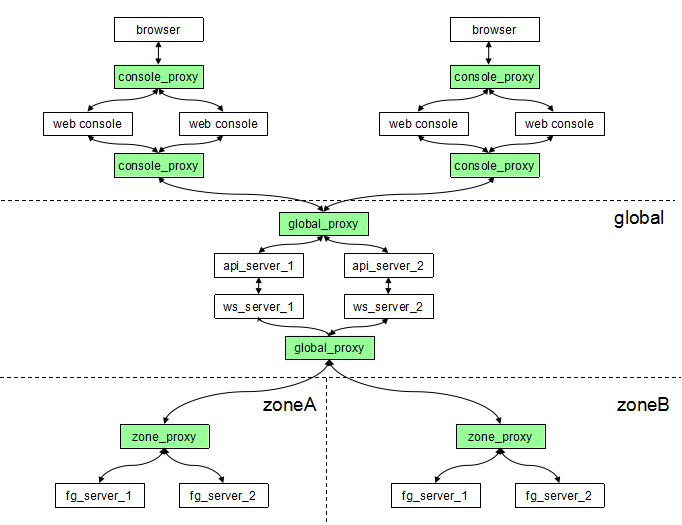
部署业务系统时，将业务系统前端挂载到虚拟LB下，并为LB绑定两个EIP（两个EIP分别属于两个VG组），通过两个EIP中的任何一个都可以对业务进行正常访问，所以在业务层需要支持通过两个IP对服务器进行访问，有两种场景，C/S客户端和B/S客户端。

* C/S客户端：需要客户端支持通过多IP访问应用服务器，通过多个IP实现业务高可用，当发现某个IP链路不通时尝试使用另外的IP链路。
* B/S客户端：需要通过F5或类似设备，对外暴露业务唯一入口，将业务流量通过轮询或其它机制传输给后端LB的两个EIP，由F5负责网络异常的处理，检查到其中一个IP链路不可用时，将流量导入另外一个IP地址，两个IP地址分别由两组VG提供，相当于将两组VG做成一个双活的Cluster。



#### 核心服务高可用

青云平台通过HAProxy机制实现所有核心服务进程的高可用和load balance管理，例如： api server所获取的console请求会先转发到ws proxy，然后发送到ws server进行处理，然后转发至 global 和 zone 之间的 proxy进行分发，请求首先到达zone内的fg proxy转发到fg server在转发到相应进程的proxy 。以下为青云平台console代理、global代理和zone代理之间的逻辑关系。

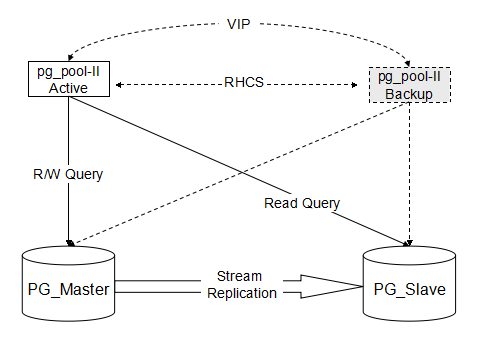


#### PG集群设计

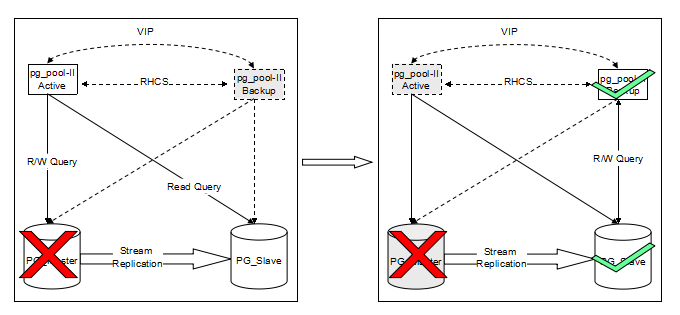
PostgreSQL是一个关系型数据库管理系统，青云平台使用PostgreSql保存青云系统数据。Global的PG保存用户数据、账户信息、配额数据等全局信息。Zone的PG保存该zone的所有资源信息。

PG\_Pool处于数据库与应用系统之间，用于实现PG cluster或分布式数据库，青云使用PG\_Pool搭建PostgreSQL集群架构，对外提供服务。Pg\_Pool采用HA架构部署，避免单点故障。

它提供以下功能：



青云配置PostgreSQL为主备模式，当PG Master故障，自动切换到PG Slave，如下图所示：



#### Zoocassa集群设计

* Zookeeper

Zookeeper 分布式服务框架用来解决分布式应用中经常遇到的一些数据管理问题，如：统一命名服务、状态同步服务、集群管理、分布式应用配置项的管理等。青云使用Zookeeper 来管理配置文件、集群管理、同步锁、Leader 选举、队列管理。

青云平台下的ZooKeeper采用分布式模式安装，配置成集群模式，集群中所有的结点作为一个整体对分布式应用提供服务，集群中每个结点之间都互相连接。ZooKeeper集群中具有两个关键的角色：Leader和Follower，在整个集群运行过程中，只有一个Leader，其他的都是Follower，如果ZooKeeper集群在运行过程中Leader出了问题，系统会采用该算法重新选出一个Leader。

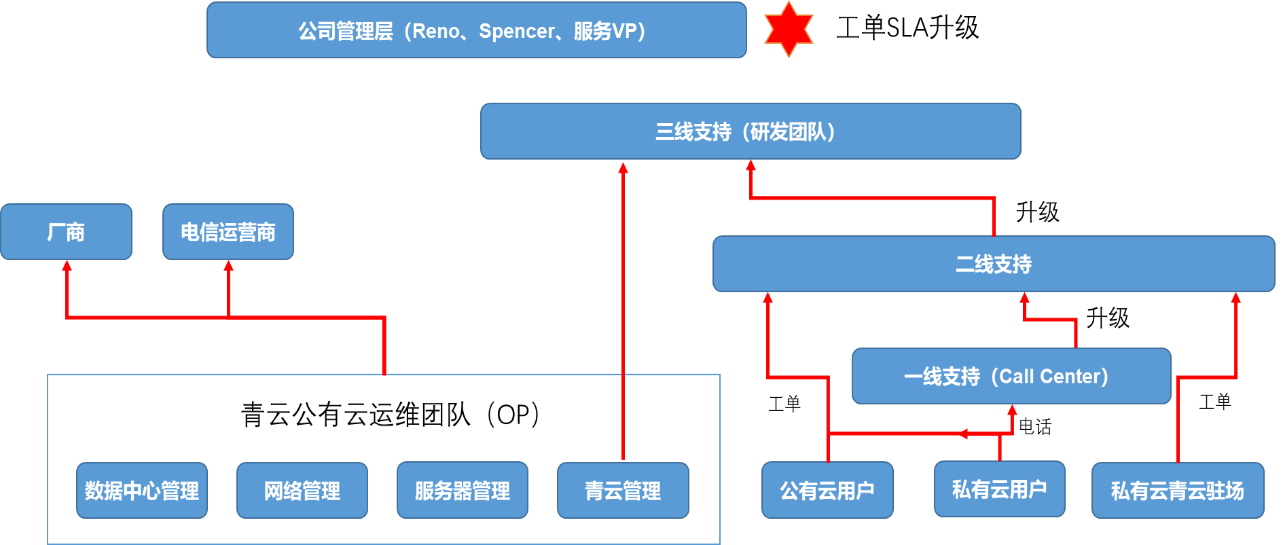
* Cassandra

Cassandra是一个分布式[NoSQL](https://zh.wikipedia.org/wiki/NoSQL)数据库系统，用于储存简单格式数据，Cassandra具有良好的[可扩展性](https://zh.wikipedia.org/wiki/可扩展性)和性能，Cassandra基于网络来处理读/写请求，同时通过网络来进行节点间的数据备份。Cassandra对副本数据的选择策略是，同机架上节点的优先于不同机架上的节点。同个数据中心的节点优于不同数据中心的节点。青云平台使用Cassandra保存青云系统的监控数据。

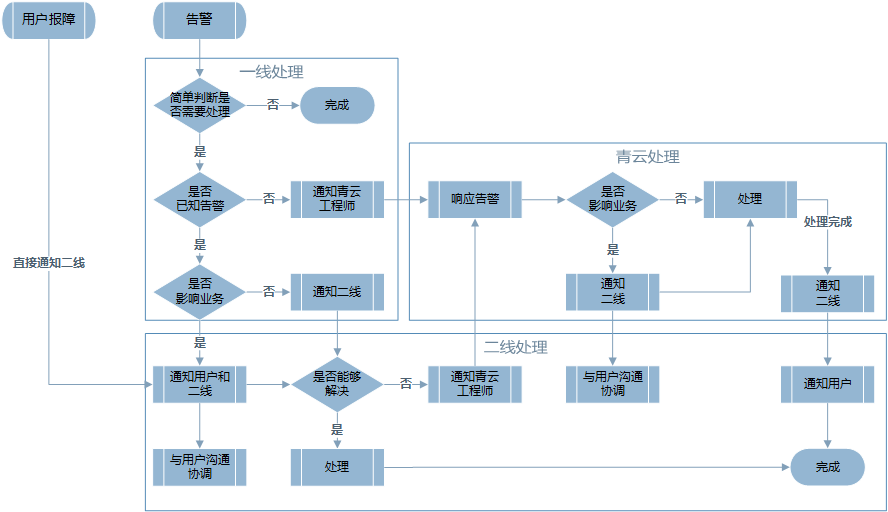
青云平台下的Cassandra配置成集群模式，统一对外提供服务。

# 青云运维管理流程

## 青云运维体系组织架构



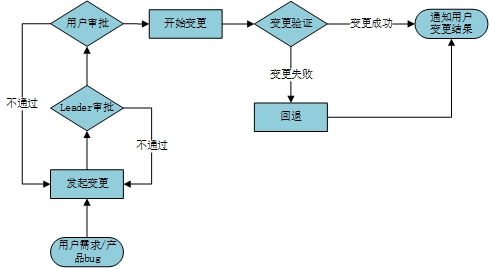
## 私有云故障处理流程



## 私有云变更流程

变更方案需要有详细步骤，尽可能详细到每一条命令，与相关产品技术负责人进行初审，之后提交客户进行终审，原则上预定操作范围即变更方案中定义的操作，不在未经许可的情况下超范围操作。操作员实施操作，复核员进行操作复核，两人共同保证变更操作不出现人为风险。

遇到紧急非预期情况，现场操作员无法独自解决时，及时申请青云技术支持远程接入，默认接入只能进行非变更类操作， 后续动作或超范围变更操作由青云变更负责人发起，青云所有参与人员进行内部评估，并由青云现场协调人与客户沟通获得操作授权之后即可进行操作



# 平台基础状态检查

## KS节点状态检查

### VG

#### 节点功能

VG为缩写，全称为Virtrual Gateway，即虚拟网关，做为青云内部与外部的边界，提供以下两个功能：

1）提供EIP服务，使vm可以通过特定的EIP地址与青云外部通信，一般情况下，在青云内资源需要与公网进行通信时使用。

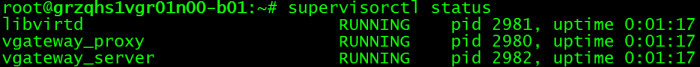
2）提供LBC（负载均衡器集群）服务，做为业务的访问入口，对业务请求进行负载均衡，此功能依赖于EIP，在创建LB时，选择公网负载均衡器，并选择一个EIP，即可以创建一个LBC服务。

#### 节点架构

VG属于ks节点，每2台为一组，架构为HA双活互备，支持无限水平扩展，一般运行在物理机上，通常情况下每个VG可承载1个C段的EIP地址，一组VG（2台）承载2个C段，如果其中一台VG宕机，其上绑定的EIP会自动由另外一台VG接管，并恢复EIP服务，VG的健康状态由zone leader节点通过heartbeat监测。

#### 状态检查

ssh登陆此节点，在命令行直接执行 supervisorctl status 检查，状态应该全部为RUNNING，且uptime持续运行时间为1分钟以上。



### Postgresql

#### 节点功能

Postgresql为青云平台的主数据库，非常重要，平台所有功能都与此数据库有关！

#### 节点架构

Postgresql属于ks节点，共2台，以主备的架构部署在2台物理机上，只有读写数据通过主节点，如果主节点宕机，需要手工将备节点升级为主节点。

#### 状态检查

* 如果数据库版本为9.3:

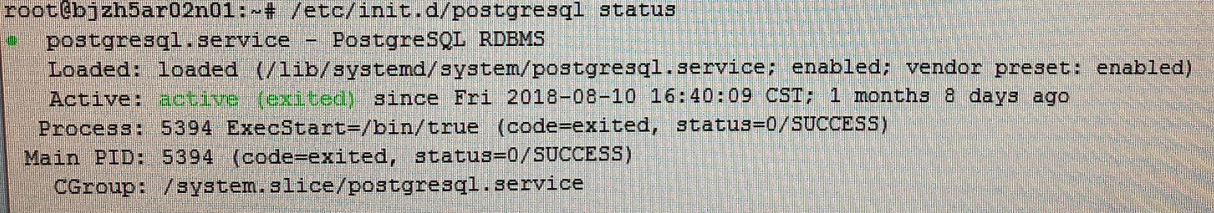
#service postgresql status

9.3/main (port 5432): online # 主库的状态

9.3/main (port 5432): online,recovery # 从库的状态

* 如果数据库版本为9.3以上版本:

systemctl status postgresql



备注：ps aux|grep wal 判断哪个是主库，哪个是从库，有 writer 和 sender的是master

root@htgt1a-pgserver0:/opt/postgresql-10.4# ps -uax|grep wal

postgres 8777 0.0 0.2 2314384 21660 ? Ss 17:04 0:00 postgres: wal writer process postgres 11158 0.0 0.1 2316708 10640 ? Ss 18:21 0:00 postgres: wal sender process postgres 10.1.0.215(52648) streaming 4/CE6BDAB0

* 检查是否有上一个整点的备份文件

|  |
| --- |
| ssh pgslave 可能pgslave找不到，可在firstbox:/pitrix/conf/settings/里查找postgresql从库是哪个节点  如果数据库版本为9.3，则备份归档目录为：  ls -arlth /var/lib/postgresql/dump\_archive  如果数据库版本为10.4，用installer4.x安装的环境都不是9.3，备份归档目录为：  ls -arlth /opt/postgresql-10.4/work/dump\_archive |

### Webservice

#### 节点功能

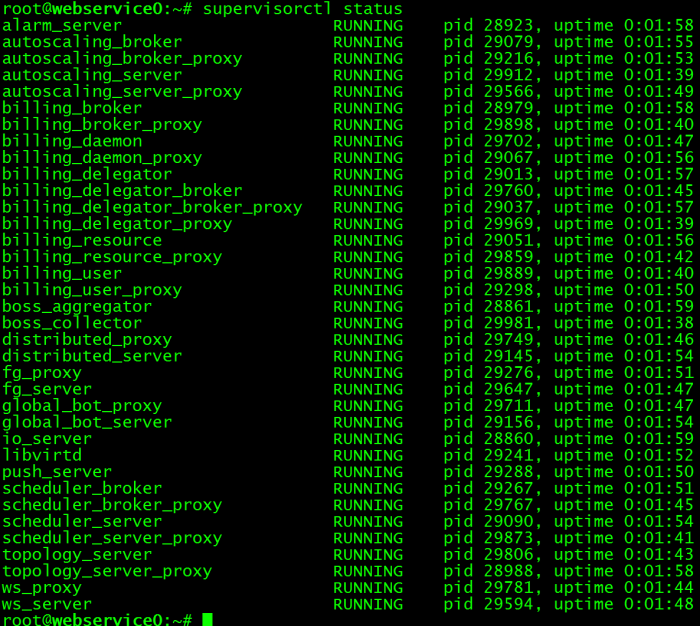
Webservice包含比较多的小组件，主要包括计费、console控制台、supervisor控制台、消息推送、BOSS、自动伸缩、全局资源锁、跨zone迁移等。

#### 节点架构

Webservice属于ks节点，共2台，做为Proxy Server的Back end节点，以虚拟机的形式运行，一般运行在非VG之外的KS物理机，单台Webservice可以满足功能需求，这里用2台满足性能及高可用的需求。

#### 状态检查

ssh登陆此节点，在命令行直接执行 supervisorctl status 检查，状态应该全部为RUNNING，且uptime持续运行时间为1分钟以上。



### Pgpool

#### 节点功能

青云平台主数据库使用PostgreSQL，PGpool是一个中间件，工作在PostgreSQL多服务器和PostgreSQL数据库客户端之间，主要是为PostgreSQL提供连接池服务。

#### 节点架构

PGpool属于ks节点，共2台，一般运行在物理机上，架构为HA主备，通过keepaleve进行心跳监测，两台PGpool节点的名称分别为pgpool0和pgpool1，共享一个vip，通过vip提供服务，平时vip配置在主节点，如果主节点宕机，在1秒左右vip会被配置到备机，完成主备切换。

#### 状态检查

在命令行直接执行 ps -ef |grep pgpool命令查看pgpool进程是否存在，大概会有400个pgpool连接池进程，使用 ps -ef |grep pgpool |wc -l统计进程数量。

C:\Users\skyshao\Pictures\20161205-60.png

### Proxy

#### 节点功能

1）提供console/supervisor/vnc的连接入口，我们所使用的 console.xxx.com、supervisor.xxx.com、api.xxx.com 这三个服务，就是解析到proxy server，由proxy server转发到后端的webservice（这三个服务实际运行在webservice上）。 通过两台Proxy Server的本地IP提供服务，两台proxy角色相同，访问任何一台都可以，在需要访问青云平台的客户机添加本地host解析，或配置dns轮询解析。

解析示例：

172.20.129.210 console.hszx.com supervisor.hszx.com #用于console/supervisor域名解析，解析到proxy0

172.20.129.211 console.hszx.com supervisor.hszx.com #用于console/supervisor域名解析，解析到proxy1

172.20.129.210 cb0grzqhs1.hszx.com #用于vnc模拟控制台域名解析，解析到proxy0

172.20.129.211 cb1grzqhs1.hszx.com #用于vnc模拟控制台域名解析，解析到proxy1

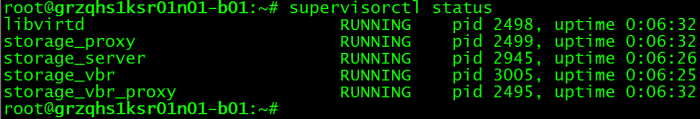
2）将青云平台的告警（包括青云系统告警及用户资源告警）发送到告警接收人，以邮件或短信形式。 通过vip对内提供告警服务，青云平台所有服务器的告警，都通过vip发送给Proxy Server，由Proxy统一将告警发送给对应的告警接收人员。

#### 节点架构

Proxy Server属于ks节点，共2台，以虚拟机的形式运行，一般分别运行在2台物理VG服务器上，架构为HA双活，通过keepaleve进行心跳监测，两台proxy节点的名称分别为proxy0和proxy1，共享一个VIP。 对于使用控制台的控制端来说，Proxy没有主备之分，访问哪台都可以。 对于青云内部模块调用Proxy，两台Proxy有主备之分，平时VIP会被配置在主节点，当主节点宕机后，VIP会飘移到备节点。

#### 状态检查

在命令行直接执行 supervisorctl status 检查，状态应该全部为RUNNING，且uptime持续运行时间为1分钟以上。



### Seed

#### 节点功能

云平台需要提供各种版本操作系统的模板，一般是windows和linux，注册到平台的模板，都上传到seed节点，在创建资源时，从seed节点调用模板，完成资源创建。

除了云平台提供的模板外，用户在console控制台基于VM（计算机与网络-主机，右键一台主机→制作成新映像）或VM的备份（存储→备份，右键一个主机的备份→制作新镜像）创建的新镜像，我们称为自有映像，也会被统一存储在seed节点。

所有模板保存在seed节点的/pitrix/images-repo目录，并进行分类，映像文件会被压缩旋转，每类模板放置于自己单独的子目录下:

1）平台提供的映像模板：子目录的名称以模板名称前两个字母命名，比如centos66x64a.tgz，前两个字母是ce，所以这个模板会被存储于/pitrix/images-repo/ce目录。

2）用户在console控制台制作的自有映像，名称格式为img-xxxxxxxx.lz4格式，子目录会以img-后面的两个字母命名，例如img-testtest.lz4文件，会被存储于/pitrix/images-repo/te目录。

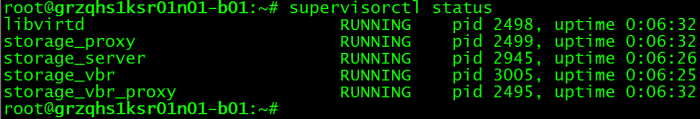
#### 节点架构

Seed属于ks节点，共2台组成集群，一般运行在物理机上，每台Seed节点可单独工作，每一台上面都有完整的镜像文件数据，官方提供的映像，会在这两台seed上分别上传一份，用户在console控制台自己制作的映像，也会将映像文件分别存储在这两台seed节点。 对于常用的映像文件，建议统一进行缓存分发，使映像文件在每台Hyper本地有一个副本，这样在创建对应映像的资源时，不再依赖Seed节点。

#### 状态检查

ssh登陆此节点，在命令行直接执行 supervisorctl status 检查，状态应该全部为RUNNING，且uptime持续运行时间为1分钟以上。

storage\_proxy、storage\_server为Seed节点的服务。



### Snapshot

#### 节点功能

Snapshot用于云平台的数据备份，与VBR节点配合使用，VBR负责将需备份数据压缩并推送到Snapshot节点，做数据恢复时VBR从Snapshot节点读取相关数据。

#### 节点架构

VBR属于ks节点，最少2台，一般运行在物理机上，属于无状态节点，每台VBR节点独立工作，最少2台起步，是考虑同时有多个任务时可以并行处理及高可用因素，在用户发起备份/恢复任务时，随时选择一台执行备份/恢复任务。

#### 状态检查

Snapshot节点无相关服务需要检查，只需要确确认snapshot目录存在即可。



### Zoocassa

#### 节点功能

Zoocassa为一组服务的统称，共包含4个服务，分别为Zookeeper、Cassandra、Redis、Memcache。

* Zookeeper

对于青云平台，Zookeeper用于保证集群稳定性、持续性，具体来说主要用于分布式集群系统的Leader选举，并对集群状态进行持续监控，如果Leader节点宕机，这时候Zookeeper会重新选出集群leader

* Cassandra

主要用于存储青云平台资源的监控数据。

* Redis

用于资源计费数据的缓存冲，降低对PostgreSQL数据库的IO压力。

* Memcache

用于缓存PostgreSQL数据库的数据，增加平台运行性能及减轻数据库的IO压力。

#### 节点架构

Zoocassa属于ks节点，以集群架构部署在3台物理机上，每台物理机上都运行了这4个服务的集群节点，最多允许坏掉一台Zoocassa，每个服务会有自己的集群状态监测。

#### 状态检查

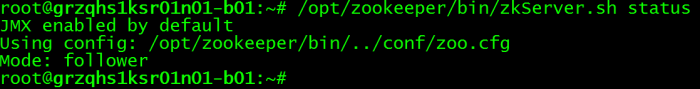
1）检查Cassandra集群，三个节点的状态应该都为UN。

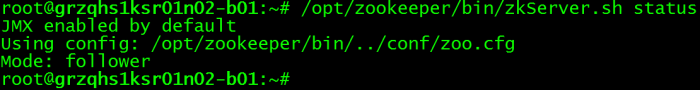
#nodetool status

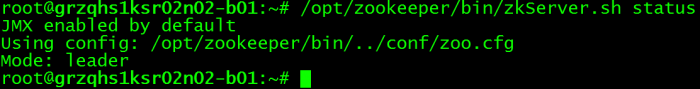


2）检查Zookeeper集群

分别登陆三台Zookeeper节点，运行/opt/zookeeper/bin/zkServer.sh status命令，应该有一个节点的Mode为leader，两个节点的状态为follower，如下：

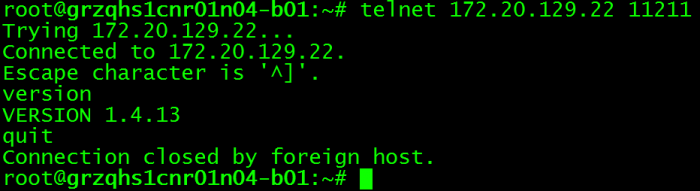






3）检查Memcache集群

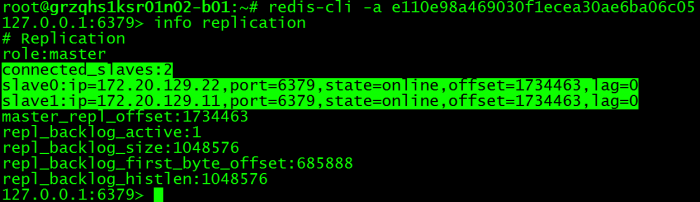
在云内任意服务器上分别telnet三台zoocassa的11211端口，看是否可以连接上，并输入version返回版本信息，quit退出。



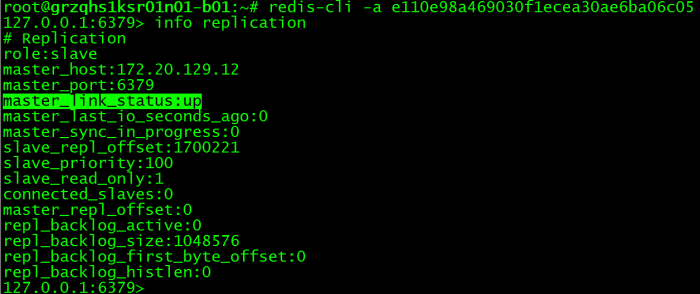
4）检查Redis集群

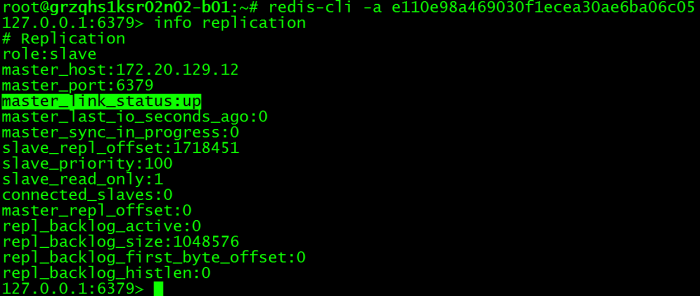
在每一台redis节点上检查节点状态

主节点状态



两个从节点状态





### Vbr

#### 节点功能

VBR的全称为Virtual Backup Recovery，为青云平台所有资源的数据备份/恢复提供引擎，VBR与Snapshot节点配合，共同完成数据备份/恢复。

备份：VBR将需要备份的数据从计算节点复制/压缩临时保存在VBR节点，最终将备份数据存储到Snapshot节点。

恢复：VBR将需要恢复的数据从Snapshot节点制取临时存储在VBR节点，最终将数据恢复到与资源相关Hyper节点。

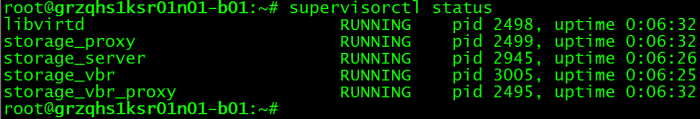
#### 节点架构

VBR属于ks节点，最少2台，一般运行在物理机上，属于无状态节点，每台VBR节点独立工作，最少2台起步，是考虑同时有多个任务时可以并行处理及高可用因素，在用户发起备份/恢复任务时，随时选择一台执行备份/恢复任务。

#### 状态检查

ssh登陆此节点，在命令行直接执行 supervisorctl status 检查，状态应该全部为RUNNING，且uptime持续运行时间为1分钟以上。

storage\_vbr、storage\_vbr\_proxy为VBR节点的服务。



## Hyper节点状态检查

### 节点功能

Hyper（计算/存储节点）：为云平台提供计算+网络+存储资源，可随业务规模横向扩展。

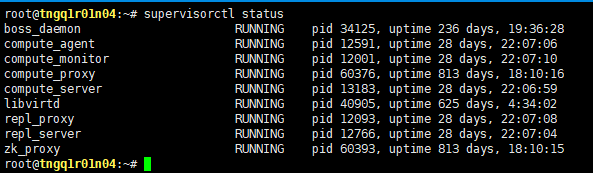
### 节点架构

青云环境计算节点以pair（组）为数据副本单元，每两台为一个pair（我们暂且称每个pair中的两台机器为A机、B机）。在SDS1.0环境下A、B互为监控、备份，即运行在A或B节点资源的数据，都有一份实时异地副本在pair内对端机器上；在SDS2.0环境下A、B互为监控，A和B的的备份可能会在集群的任一节点上。

### 状态检查

#### 服务检查

ssh登陆，在命令行直接执行 supervisorctl status 检查，状态应该全部为RUNNING，且uptime持续运行时间为1分钟以上。



#### SDS状态检查

针对SDS1.0和2.0的不同，巡检的方式有所差异。

##### SDS1.0

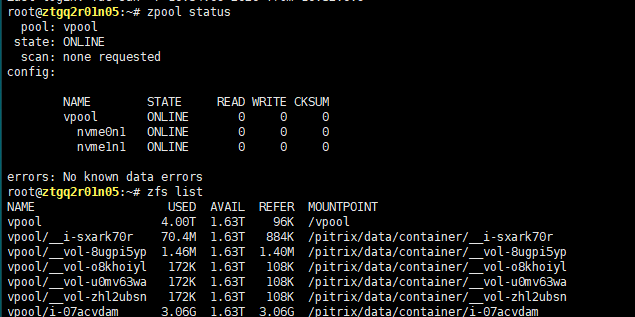
查看drbd状态：drbd-overview



##### SDS2.0

查看存储池状态 ：zpool status

列出存储池中的文件系统：zfs list



# 平台基础状态维护

## KS节点关机流程

这里指的KS节点，是除了Hyper以外的所有节点。

这里只做简要说明，具体操作时需要考虑关闭KS机器的影响，关于具体KS节点功能说明及关机影响，需要看一下对应KS节点的功能说明、应急预案和迁移方案，有助于理解各KS节点，会明白关机维护应该如何做。

1）如果物理KS节点上，运行了VM KS节点的虚拟机，需要先将VM关机。

2）如果VG上有EIP，需要先将EIP迁移走。（迁移EIP可参考[《VG节点迁移》](#_VG节点迁移)）

3）注意检查是否有其它KS节点把VG做为了网关。

4）尽量在夜间业务低谷时操作。

## Hyper节点关机流程

关机维护前涉及资源的在线迁移， 在线迁移会导致资源有1-5分钟左右的中断，所以迁移前需要与客户沟通确认操作窗口。

1）在Supervisor上，将此Hyper状态置为Standby，使Hyper不再接受创建新资源的任务。

2）将Hyper上运行的所有资源迁移走，保证/pitrix/data/contrainer下没有任何资源文件。

登录webservice0/1，进入cli目录

# ssh webservice0

# cd /pitrix/cli

migrate-instances #此命令用于迁移单个或多个vm。

migrate-hyper-node #此命令用于迁移此Hyper上所有资源。

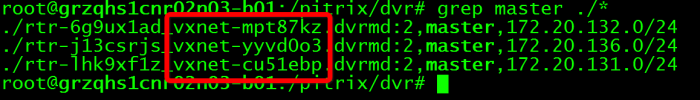
3）如果是SDN2.0，需要将vxnet（dvr master）迁移走，如不做特殊说明，这里vxnet和dvr master是一个东西。

查看此hyper上是否有dvr master.

# cd /pitrix/dvr

# grep master ./\*

如果有输出，说明此hyper上运行了dvr master服务，如果没有输出，说明没有，不用操作。



迁移vxnet使用以下命令：

# migrate-vxnets

迁移后，从云外访问一下对应的基础网络中vm，确认迁移后网络是好的。

4）将pair节点的compute\_agent服务暂时停掉，此进程用于监控pair节点的健康状态及触发自动迁移，pair物理机维护完成后，再将此服务启动。

查看服务状态

# supervisorctl status

停止compute\_agent服务

# supervisorctl stop compute\_agnet

启动compute\_agent服务

# supervisorctl start compute\_agnet

5）关闭Hyper

Icas架构的sds1.0节点，在使用时要特别注意按以下流程关机：

supervisorctl stop all

swapoff -a

umount /pitrix/data/container

drbdadm down tr02n26

drbdadm down tr02n27

shutdown -h now

根据输出的提示进行关机操作。

## 平台升级

平台升级请参考《IAAS平台版本升级方案》

## 平台扩容

平台扩容包含KS节点扩容和hyper节点扩容，可参考《IAAS平台扩容方案》

## 平台关闭

平台关闭顺序与平台架构有关，架构分为SANC架构与非SANC架构，SANC架构下细分为融合部署与独立部署两种，不同架构下平台关闭流程如下：

1. **在SANC或非SANC架构下，关闭平台前，需确认：**

* 确认没有任务在执行。
* 所有用户资源已关闭。

1. **对于非SANC架构，正常按照本文档5.5.1-5.5.5章节描述关闭。**
2. **对于SANC架构，分为两种情况。**

* NeonSAN与IAAS融合部署。

1. 先关闭IAAS所有服务，但不关闭物理机，参考本文档5.5.1-5.5.3。
2. 关闭NeonSAN管理服务及存储服务，参考《QingStor NeonSAN管理员手册》第3章节。
3. 关闭IAAS计算节点及管理节点物理机，参考本文档5.5.4-5.5.5。

* NeonSAN与IAAS分别独立部署。

1. 先关闭IAAS所有服务，包括物理机，参考本文档5.5.1-5.5.5。
2. 关闭NeonSAN服务，参考《QingStor NeonSAN管理员手册》第3章节。

### 将所有计算节点状态设置为故障（faulty）

确认用户资源已经关闭，才能进行操作

登陆supervisor或boss控制台，二选一。

* 在supervisor中，通过机器人菜单（boss中为hypenode）页面修改计算节点状态为faulty （故障）。



* 在boss中，基础设施🡪计算节点，选中节点🡪标记状态，选择“不可用”状态。





### 禁用灾难检测和恢复机制

登录firstbox，备份并修改/pitrix/conf/variables/global/server.yaml 文件，

将compute\_server 下面的 disable\_hyperpair\_rescue 参数，配置成1，并同步到所有节点。这样会禁用灾难检测和恢复机制，先调整此配置再关机，可以避免开机的时候可能有灾难恢复被意外触发的情况发生。

配置文件同步方法如下:

cd /pitrix/upgrade/

./build\_global\_conf.sh

./update\_nodes.sh all pitrix-global-conf

### 关闭控制台服务

控制台一般和API服务装在一起，所以需要登陆API服务节点，并关闭控制台，以防止有环境在接收新的任务。API服务位于webservice0,webservice1上. 关闭步骤如下，以webservice0为例：

登陆webservice0 (webservice1也需要如此操作)

# ssh webservice0

关闭控制台服务

# /etc/init.d/apache2 stop

# /etc/init.d/nginx stop

退出

#exit

### 关闭计算节点

在做以下步骤前，一定要确保用户所有的资源已经关闭。

1）依次登陆计算节点，关闭服务,以一个节点为例，其他相同。

# ssh jq1r01n05

# supervisorctl

suprervisorctl> stop all

suprervisorctl> quit

也可直接到firstbox上执行批量命令 /pitrix/upgrade/exec\_nodes.sh hyper 'supervisorctl stop all'

2）卸载swap

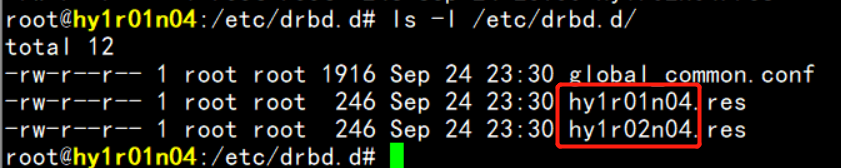
swapoff -a

3）卸载数据盘目录

umount /pitrix/data/container

4）停掉 drbd

命令 drbdadm down xxx ，执行两次停掉两个盘 xxx 根据情况替换，因为每台计算节点的配置不一样所以需要手动登录到每台上区执行。xxx的具体替换值可通过ls -l /etc/drbd.d/查看，如下图就需要在这台执行drbdadm down hy1r01n04 和drbdadm down hy1r02n04



5）关闭机器。

# shutdown -h now 或者 sudo shutdown -h now 如果还是提示指定目录下有文件不让关机则可以直接执行 init 0

### 关闭管理节点

#### 关闭虚机类型的管理节点

|  |  |  |
| --- | --- | --- |
| 类型 | 主机名 | Ip |
| 虚拟机 | proxy0 | 21.122.168.220 |
| 虚拟机 | proxy1 | 21.122.168.221 |
| 虚拟机 | webservice0 | 21.122.168.222 |
| 虚拟机 | webservice1 | 21.122.168.223 |
| 虚拟机 | dnsmaster0 | 21.122.168.33 |
| 虚拟机 | dnsmaster1 | 21.122.168.34 |

|  |
| --- |
| # ssh 到对应节点上,先关闭proxy，其它顺序无要求  # supervisorctl stop all  # shutdown -h now 或者 sudo shutdown -h now 如果还是提示指定目录下有文件不让关机则可以直接执行 init 0 |

#### 关闭物理机类型的管理节点

|  |  |  |
| --- | --- | --- |
| 主机名 | 服务器角色 | 物理IP |
| bocd1r01n00 | proxy0(vm)/firstbox(vm)/zoocassa0/pgpool0 | 21.122.168.40 |
| bocd1r01n01 | zoocassa1/pgserver0/webservice0(vm)/dns-master(vm) | 21.122.168.41 |
| bocd1r02n00 | seed0/vbr0/snapshot0/zoocassa2/pgserver1/proxy1(vm) | 21.122.168.50 |
| bocd1r02n01 | seed1/vbr1/snapshot1/webservice1(vm)/pgpool1/dns-slave(vm) | 21.122.168.51 |

上表为每台管理节点上承载的服务，可通过此表对照哪个服务运行在哪台物理机上。

##### 关闭zoocassa服务

包括cassandra、zookeeper、memcache、redis

分别ssh到三台zoocassa节点上操作

service cassandra stop

/opt/zookeeper/bin/zkServer.sh stop

service memcached stop

service redis\_6379 stop

redis-cli -p 26379 shutdown

##### 关闭pgpool

远程到对应节点

service pgpool2 stop

##### 关闭pgserver

远程到对应节点

service postgres stop

##### 所有物理机管理节点

|  |
| --- |
| # ssh 到对应节点上  # supervisorctl stop all  # shutdown -h now 或者 sudo shutdown -h now 如果还是提示指定目录下有文件不让关机则可以直接执行 init 0 |

## 平台启动

平台启动，同样需要考虑SANC架构与非SANC架构。

青云系统内部的所有服务都是自启动的，所以一般情况下，只需要管理员根据架构不同按顺序启动物理机，物理机启动后登陆OS检查服务状态。

1. **对于非SANC架构，正常按照本文档5.6.1-5.6.4章节描述启动。**
2. **对于SANC架构，分为两种情况。**

* NeonSAN与IAAS融合部署。

1. 启动所有物理机。
2. 检查NeonSAN服务，参考《QingStor NeonSAN管理员手册》第3章节。
3. 确认NeonSAN运行正常后，检查IAAS服务状态，参考本文档5.6.1-5.6.6。

* NeonSAN与IAAS分别独立部署。

1. 启动NeonSAN服务，参考《QingStor NeonSAN管理员手册》第3章节。
2. 确认NeonSAN服务正常后，启动IAAS物理机及检查服务，参考本文档5.6.1-5.6.6。

### 检查节点时钟同步

重启后需检查所有节点时钟是否同步，ntp服务是否工作正常。

ssh firstbox

cd /pitrix/upgrade/

./exec\_nodes.sh all 'date'

./exec\_nodes.sh all 'ntpq -np'

### 检查所有管理节点的状态

|  |
| --- |
| ssh firstbox  cd /pitrix/upgrade/  检查supervisorctl服务，确认所有服务状态为running，且运行时间为1分钟以上。  ./exec\_nodes.sh non-hyper 'supervisorctl status' |

### 检查所有hyper节点的状态，包括

ssh firstbox

cd /pitrix/upgrade/

./exec\_nodes.sh hyper 'supervisorctl status'

./exec\_nodes.sh hyper 'drbd-overview'

### 将计算节点状态改

登陆supervisor或boss控制台，通过机器人菜单（boss中为hypenode）页面修改计算节点状态为为active或standby（恢复hyper的原有状态），最终可用需要状态为active活跃。



### 登陆console进行测试

1) 登陆iaas平台测试创建所有资源正常

2) 测试iaas平台资源迁移是否正常，对测试的资源进行迁移

3) 登录console界面，验证console界面功能可用性

### 恢复灾难检测和恢复机制

等环境物理机全部开机并恢复后再登录firstbox，修改/pitrix/conf/variables/global/server.yaml 文件，将compute\_server 下面的 disable\_hyperpair\_rescue 参数改成0，再同步到所有节点，重启compute\_agent，方法如下：

|  |
| --- |
| cd /pitrix/upgrade/  ./build\_global\_conf.sh  ./update\_nodes.sh all pitrix-global-conf  ./exec\_nodes.sh hyper ‘supervisorctl restart compute\_agent’ |

重新启用灾难恢复机制。

# 平台用户管理

## BOSS用户管理

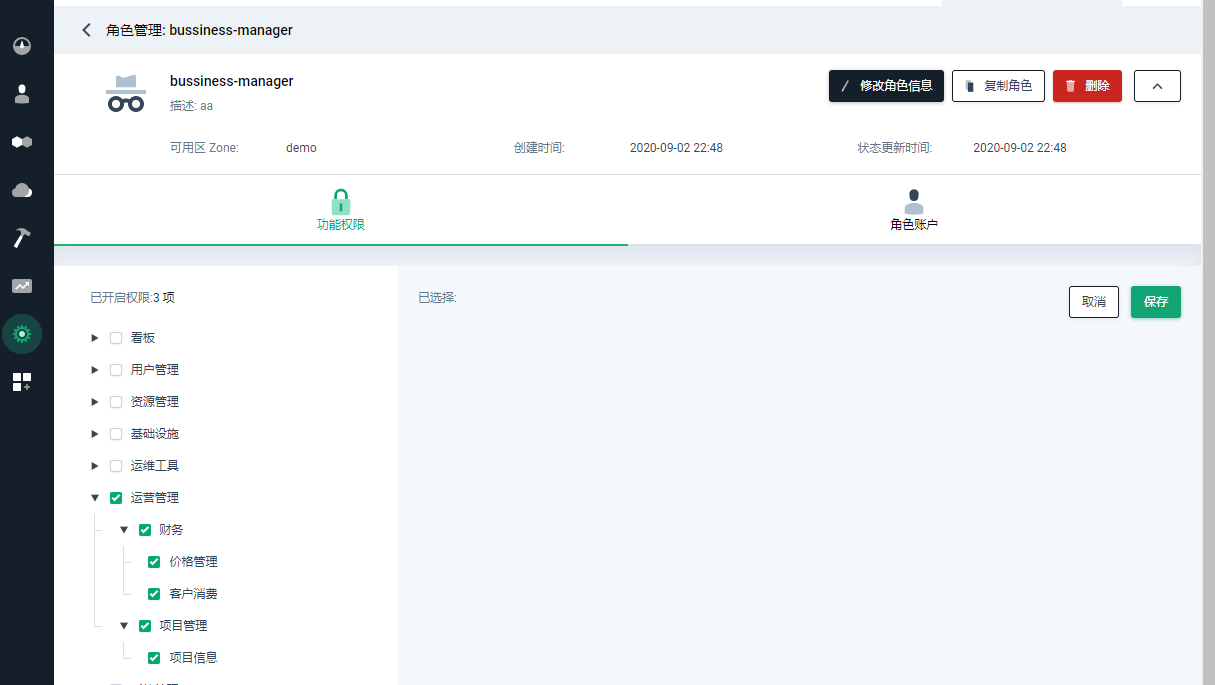
### BOSS概述

BOSS：统一运维管理平台。替代了原supervisor的功能，用于管理员管理、监控、展示青云平台资源。

### BOSS角色管理

超级管理员：该角色拥有所有页面权限，供最高级别管理人员使用；

自定义角色：可创建自定义角色并关联相应的功能权限；



### BOSS用户管理

BOSS平台初始会提供一个管理员用户，一般为boss*@domain*，初始密码为zhu88jie。

BOSS的用户和console的用户是相互独立。BOSS的用户位于BOSS页面的“系统管理”-“权限管理”下的账户管理。创建新的BOSS用户时选择相应的角色即可。



## 控制台用户

### 控制台用户概述

控制台用户也是supervisor用户，在青云平台里用户的权限是通过角色来区分的，不同的角色拥有不同的权限。默认的角色如下：

* 全球管理员（可管理与global ks相连的所有资源）
* 控制台管理员（只能管理console id跟自己匹配的资源）
* 合作伙伴（可以在appcenter创建APP，以及部分开放给合作伙伴的API接口）
* 普通用户

青云平台初始会提供一个全球管理员用户，一般为admin@*domain*，初始密码为zhu1241jie。

### 用户管理

用户的管理一般包含用户的创建、权限的分配、配额管理。

创建用户时要注意“**付费类型**”选项。针对公有云用户默认采用的是 “预付费”，即当用户欠费后会暂停资源然后删除资源；私有云默认采用的是“后付费”，即只记录资源产生的费用。

#### 通过supervisor管理

青云平台管理员可以登录青云supervisor管理员页面。在页面的左侧导航菜单下，选择“用户”,supervisor右侧主显示区域会显示用户详情页。



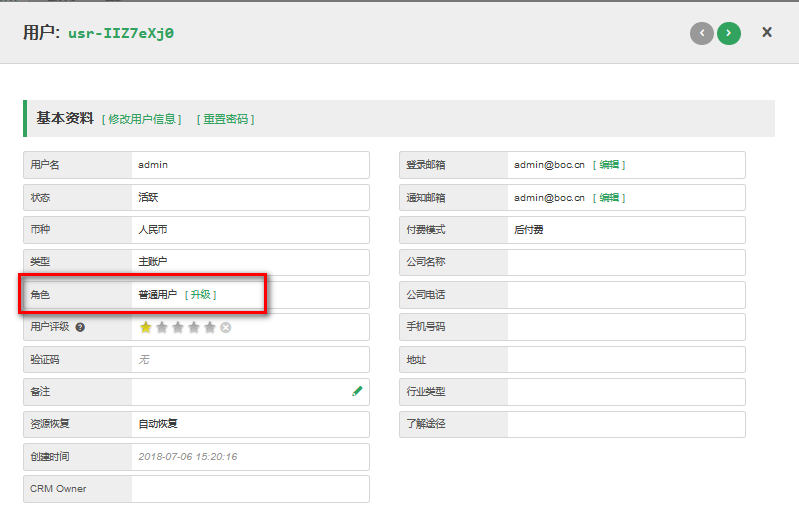
* 创建用户

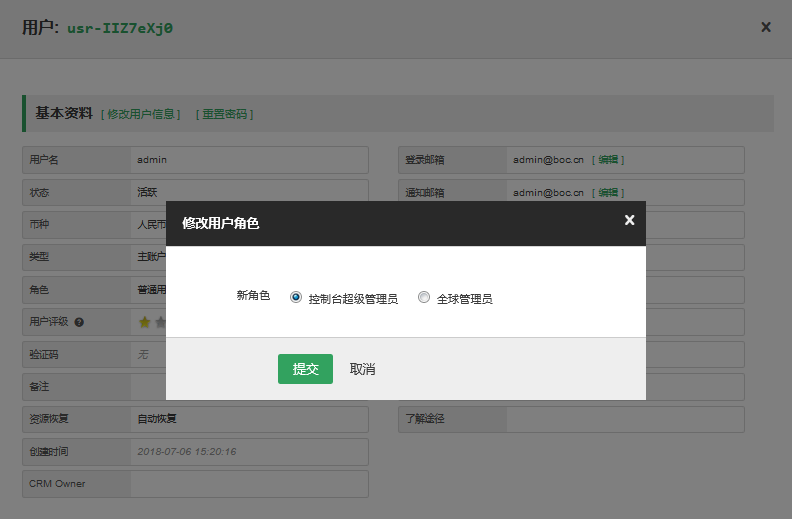
管理员在右侧主显示区域上方功能按钮区域，选择“创建用户”，系统会弹出“创建用户”窗口。根据页面提示的信息输入或勾选相应的信息（根据实际情况选择付费模式）。



* 修改用户角色：

单击用户显示用户详细信息，在“角色”栏中点击升级可根据提示将用户角色升级为“控制台超级管理员”或“全球管理员”。





* 用户配额管理

青云平台管理员可以登录青云supervisor管理员页面。在页面的左侧导航菜单下，选择“用户配额”,supervisor右侧主显示区域会显示“用户配额”详情页。



管理员在右侧主显示区域，双击某一用户，系统会弹出“用户配额”窗口。







管理员在修改用户配额以后，需要单击“修改”摁钮进行确认。

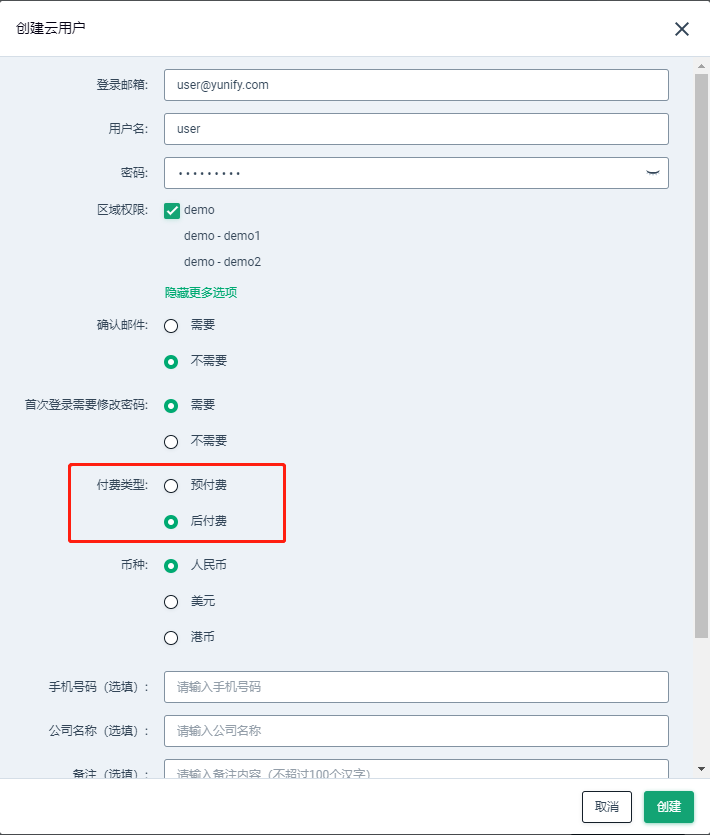
#### 通过BOSS管理

可通过登录BOSS页面，打开“用户管理”-“云用户”对控制台用户进行管理。



* 创建用户

根据页面提示的信息输入或勾选相应的信息（根据实际情况选择付费类型）。



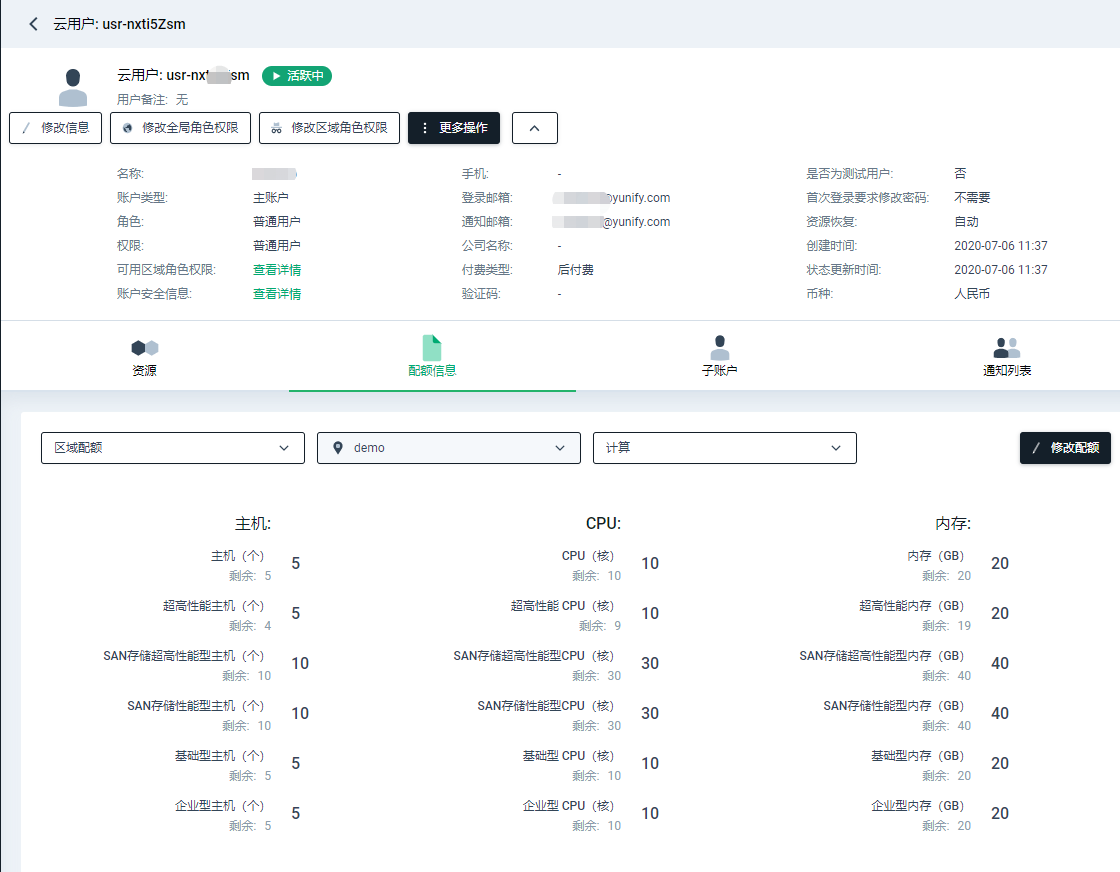
* 修改用户角色

用户创建完成后，可选择修改用户的角色权限（默认为普通用户）。



* 用户配额管理

选中相应的云用户，在配额信息里可修改相应的配额。



# 平台日常维护

## 日常维护命令

### 切换zone leader

在zone leader节点，重启compute\_server服务，leader节点会被重新选举。

如何确定哪台Hyper为zone leader。

随便找一台hyper，grep leader compute\_server.log，看最后一行或者查看cat /pitrix/run/zone\_leader文件。

### 数据库操作工具exec\_sql

偶尔需要对数据库进行操作，风险极高，使用此辅助命令可以降低数据库误操作的概率，为了安全。

工具命令：exec\_sql

工具位置：webservice0或webservice1主机的/pitrix/lib/pitrix-scripts目录

工具作用：执行一条sql语句，只允许select、insert、update操作，且update操作一次只能update一条记录。

命令格式：

./exec\_sql -d zone -c "在这里写sql语句"

命令举例：

查看目录所有正在运行中的vm

./exec\_sql -d zone -c "select \* from instance where status = ‘running’"

### 批量执行操作exec\_nodes.sh

工具命令：exec\_nodes.sh

工具位置：firstbox主机的/pitrix/upgrade目录

工具作用：批量在指定的节点上执行某个命令

命令格式：

/pitrix/upgrade/exec\_nodes.sh node\_name “your command”

注：node\_name可以是一个hostname，也可以是hostname\_group，hostname\_group的定义在/pitrix/upgrade/node目录中，每个文件代表一个hostname\_group，文件中的内容代表hostname\_group所包含的主机清单，hostname\_group包括：

all、hyper、hyper-hc、hyper-hp、hyper-perf、hyper-s3、nonhyper、proxy、seed、vbr、vgateway、virt-nodes、vxnet-nodes、webservice、zoocassa

命令举例：

/pitrix/upgrade/exec\_nodes.sh all “ls -l” # 代表在所有节点上执行ls –l命令。

### 配置同步

本说明中所有操作都是在firstbox中执行。

青云平台的global-conf、hosts、alert-agent配置维护方式，是统一在firstbox中进行修改，然后将修改后的配置同步到所有节点。

在对global-conf、hosts、alert-agent等配置做修改后，需要同步到所有节点，在同步之前，需要将配置打包，build\_global\_conf.sh命令就是用来打包的，打包后再通过firstbox:/pitrix/upgrade/update.sh命令进行同步。

以下情况时需要将配置同步到所有节点：

当firstbox:/pitrix/conf/variables/global\_conf/global/下的配置文件有修改时，需要更新global conf。

当firstbox:/pitrix/conf/variables/hosts文件有修改或firstbox:/pitrix/conf/settings下有新增节点时，需要更新hosts。

firstbox:/pitrix/conf/packages/pitrix-alert-agent/alert-agent文件有修改时。

#### build\_global\_conf.sh命令说明

命令路径：

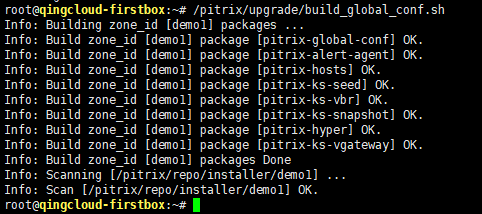
firstbox:/pitrix/upgrade/build\_global\_conf.sh

命令格式：

build\_global\_conf.sh命令无附加参数，直接执行。

举例：

在firstbox中，直接执行/pitrix/upgrade/build\_global\_conf.sh，正常输出如下：



附：build\_global\_conf.sh是安全的更新工具,因为它没有任何覆盖的操作，只做四件事，在本文档中，前两条对我们有用。

1）把现有的fb上的/pitrix/conf/variables/global\_conf/global/里的配置文件打包成pitrix-global-conf

2）根据/pitrix/conf/settings下的节点信息,调用\_sync\_hosts.sh 生成新的pitrix-hosts包

3）根据/pitrix/conf/settings下的节点信息,调用\_sync\_hyper.sh生成注册hyper节点必须的sql语句和新的pitrix-hyper包

pitrix-hyper包存在的意义主要在于两点：

a）提供了安装一个hyper所需要的所有的依赖包 (/pitrix/conf/packages/pitrix-hyper/control) 这个依赖会根据是否sdn1.0/2.0而有所不同,sdn2.0会多两个包(pitrix-dep-kernel-patch, pitrix-dep-iproute2)

b）\_sync\_hyper.sh 会根据setting下的hyper节点信息生成用户注册hyper到db里的sql语句 (/pitrix/bin/sync.d/.deb/hyper/pitrix/hyper/data/)

4）刷新/pitrix/repo/indep/下面的包的索引 (scan package)

#### update.sh命令说明

命令路径：

firstbox:/pitrix/upgrade/update.sh

命令格式：

/pitrix/upgrade/update.sh <node\_name> <comp\_name>

<node\_name>可以是一个hostname，也可以是hostname\_group，hostname\_group的定义在/pitrix/upgrade/node目录下，每个文件代表一个hostname\_group，文件中的内容代表hostname\_group所包含的主机清单。

<comp\_name>是节点的角色名称，角色名称的定义在/pitrix/upgrade/comp目录下，每个角色文件中包含了更新这个角色时所需要更新的所有deb包名称，分以下几类。

alert-agent、global-conf、global-webservice、hosts、hyper、proxy、seed、vbr、vgateway、zone-webservice

常用举例：

更新所有节点的global\_conf配置，使用以下命令。

[update.sh](http://update.sh/) all global-conf

更新所有节点的hosts

[update.sh](http://update.sh/) all hosts

更新所有节点的alert-agent

更新所有节点的alert-agent使用以下命令。

[update.sh](http://update.sh/) all alert-agent

以上三个举例，如果只想更新某个节点，或某类节点，可以把all换成主机名，或hostname\_group名称。

更多的命令帮助信息可以执行 /pitrix/upgrade/update.sh -h查看。

## 故障处理

### DRBD故障修复

#### 如何查看drbd状态

可以通过 drbd-overview 命令查看本机drbd各盘状态，命令输出结果示例如下

0:slcy1r01n06/0 Connected Primary/Secondary UpToDate/UpToDate /pitrix/data/container xfs 4.1T 485G 3.7T 12%

1:slcy1r01n04/0 Connected Secondary/Primary UpToDate/UpToDate

所有数据结果均为本机上drbd盘的当前状态，可以看到本机有两块drbd盘，第一块盘作为本机数据盘使用，第二块盘作为pair数据盘的备盘使用。

以下分别解释输出结果中每个字段含义：

0:slcy1r01n06/0 drbd盘id

Connected 连接状态

Primary/Secondary 本地盘角色/对端盘角色

UpToDate/UpToDate 本地同步状态/对端盘同步状态

/pitrix/data/container 挂载点（盘被挂载才会显示）

xfs 文件系统（盘被挂载才会显示）

4.1T 总容量（盘被挂载才会显示）

485G 已用容量（盘被挂载才会显示）

3.7T 剩余容量（盘被挂载才会显示）

12% 使用率（盘被挂载才会显示）

状态查看  
[root@chy database]# drbd-overview

0:r0/0 Connected Primary/Secondary UpToDate/UpToDate /database ext4 93M 1.6M 85M 2%

[root@chy database]# cat /proc/drbd

version: 8.4.10-1 (api:1/proto:86-101)

GIT-hash: a4d5de01fffd7e4cde48a080e2c686f9e8cebf4c build by mockbuild@, 2017-09-15 14:23:22

0: cs:Connected ro:Primary/Secondary ds:UpToDate/UpToDate C r-----

ns:6 nr:24 dw:30 dr:1385 al:2 bm:0 lo:0 pe:0 ua:0 ap:0 ep:1 wo:f oos:0

cs: connect state   
ro:表示角色信息   
ds: 磁盘状态信息Inconsistent/UpToDate   
ns/nr:网络发送/接收的数据包信息  
dw/dr:设备读写信息

[root@chy database]# drbdadm cstate r0 //资源的连接状态（r0是资源）

Connected

资源的连接状态；一个资源可能有以下连接状态中的一种   
StandAlone 独立的：网络配置不可用；资源还没有被连接或是被管理断开（使用 drbdadm disconnect 命令），或是由于出现认证失败或是脑裂的情况   
Disconnecting 断开：断开只是临时状态，下一个状态是StandAlone独立的   
Unconnected 悬空：是尝试连接前的临时状态，可能下一个状态为WFconnection和WFReportParams   
Timeout 超时：与对等节点连接超时，也是临时状态，下一个状态为Unconected悬空   
BrokerPipe：与对等节点连接丢失，也是临时状态，下一个状态为Unconected悬空   
NetworkFailure：与对等节点推动连接后的临时状态，下一个状态为Unconected悬空   
ProtocolError：与对等节点推动连接后的临时状态，下一个状态为Unconected悬空   
TearDown 拆解：临时状态，对等节点关闭，下一个状态为Unconected悬空   
WFConnection：等待和对等节点建立网络连接   
WFReportParams：已经建立TCP连接，本节点等待从对等节点传来的第一个网络包   
Connected 连接：DRBD已经建立连接，数据镜像现在可用，节点处于正常状态   
StartingSyncS：完全同步，有管理员发起的刚刚开始同步，未来可能的状态为SyncSource或PausedSyncS   
StartingSyncT：完全同步，有管理员发起的刚刚开始同步，下一状态为WFSyncUUID   
WFBitMapS：部分同步刚刚开始，下一步可能的状态为SyncSource或PausedSyncS   
WFBitMapT：部分同步刚刚开始，下一步可能的状态为WFSyncUUID   
WFSyncUUID：同步即将开始，下一步可能的状态为SyncTarget或PausedSyncT   
SyncSource：以本节点为同步源的同步正在进行   
SyncTarget：以本节点为同步目标的同步正在进行   
PausedSyncS：以本地节点是一个持续同步的源，但是目前同步已经暂停，可能是因为另外一个同步正在进行或是使用命令(drbdadm pause-sync)暂停了同步   
PausedSyncT：以本地节点为持续同步的目标，但是目前同步已经暂停，这可以是因为另外一个同步正在进行或是使用命令(drbdadm pause-sync)暂停了同步   
VerifyS：以本地节点为验证源的线上设备验证正在执行   
VerifyT：以本地节点为验证目标的线上设备验证正在执行  
  
资源角色  
[root@chy database]# drbdadm role r0  
Primary/Secondary  
Parimary 主：资源目前为主，并且可能正在被读取或写入，如果不是双主只会出现在两个节点中的其中一个节点上   
Secondary 次：资源目前为次，正常接收对等节点的更新  
Unknown 未知：资源角色目前未知，本地的资源不会出现这种状态  
  
硬盘状态  
[root@chy database]# drbdadm dstate r0 //r0是资源  
UpToDate/UpToDate  
  
本地和对等节点的硬盘有可能为下列状态之一：   
Diskless 无盘：本地没有块设备分配给DRBD使用，这表示没有可用的设备，或者使用drbdadm命令手工分离或是底层的I/O错误导致自动分离   
Attaching：读取无数据时候的瞬间状态   
Failed 失败：本地块设备报告I/O错误的下一个状态，其下一个状态为Diskless无盘   
Negotiating：在已经连接的DRBD设置进行Attach读取无数据前的瞬间状态   
Inconsistent：数据是不一致的，在两个节点上（初始的完全同步前）这种状态出现后立即创建一个新的资源。此外，在同步期间（同步目标）在一个节点上出现这种状态   
Outdated：数据资源是一致的，但是已经过时   
DUnknown：当对等节点网络连接不可用时出现这种状态   
Consistent：一个没有连接的节点数据一致，当建立连接时，它决定数据是UpToDate或是Outdated   
UpToDate：一致的最新的数据状态，这个状态为正常状态

#### Unconfigured状态的处理

示例

root@slcy1r01n04:~# drbd-overview

0:slcy1r01n04/0 WFConnection Primary/Unknown UpToDate/DUnknown /pitrix/data/container xfs 4.1T 322G 3.8T 8%

1:slcy1r01n06/0 Connected Secondary/Primary UpToDate/UpToDate

root@slcy1r01n06:~# drbd-overview

0:slcy1r01n06/0 Connected Primary/Secondary UpToDate/UpToDate /pitrix/data/container xfs 4.1T 485G 3.7T 12%

1:slcy1r01n04/0 Unconfigured . .

可以看到06上的从盘状态为Unconfigured,这说明该盘处于down的状态，可以用 “drbdadm up 盘id” 命令来修改状态，执行

root@slcy1r01n06:~# drbdadm up slcy1r01n04

之后可以看到盘已经up，并已经开始同步了

root@slcy1r01n06:~# drbd-overview

0:slcy1r01n06/0 Connected Primary/Secondary UpToDate/UpToDate /pitrix/data/container xfs 4.1T 485G 3.7T 12%

1:slcy1r01n04/0 SyncTarget Secondary/Primary Inconsistent/UpToDate

[>....................] sync'ed: 0.9% (13448/13556)M

#### 主、备盘角色正确，主盘WFConnection，备盘StandAlone 状态的处理

示例

root@cms1r01n02:~#drbd-overview

0:cms1r01n02/0 SyncSource Primary/Secondary UpToDate/Inconsistent C r----- /pitrix/data/container xfs 19T 209G 18T 2%

[==========>.........] sync'ed: 57.9% (8120/19272)Mfinish: 0:03:57 speed: 34,976 (25,716) K/sec

1:cms1r01n03/0 StandAlone Secondary/Unknown UpToDate/DUnknown r-----

root@cms1r01n03:~#drbd-overview

0:cms1r01n03/0 WFConnection Primary/Unknown UpToDate/DUnknown C r----- /pitrix/data/container xfs 19T 107G 19T 1%

1:cms1r01n02/0 SyncTarget Secondary/Primary Inconsistent/UpToDate C r-----

[===========>........] sync'ed: 62.3% (7272/19272)Mfinish: 0:03:31 speed: 35,224 (26,144) want: 71,720 K/sec

可以看到 cms1r01n03/0 这组盘没有同步，主盘角色为Primary，备盘角色为Secondary，说明主、备盘角色是正确的，不需要调整，这时只要丢弃备盘中的数据即可使之同步。命令示例如下

root@cms1r01n02:~#drbdadm connect --discard-my-data cms1r01n03

#### 主、备盘角色不正确的处理

主、备盘角色不正确有以下几种情况，修改盘角色之前，需要先将盘卸载掉

##### 主备盘都为Primary/Unknown

处理方法，将备盘状态修改正确

root@备盘节点:~# drbdadm secondary 备盘id

##### 主备盘都为Secondary/Unknown

处理方法，将主盘状态修改正确

root@主盘节点:~# drbdadm primary --force 主盘id

##### 主盘状态为Secondary/Unknown，备盘状态为Primary/Unknown

这种情况要保证备盘数据都已经迁移走，先调整主盘角色为Primary，再调整备盘角色为Secondary

root@主盘节点:~# drbdadm primary --force 主盘id

root@备盘节点:~# drbdadm secondary 备盘id

#### drbd盘修改状态失败，提示Device is held open by someone 或 busy 的处理

调整盘状态失败，问题示例如下

root@llhd1r02n05:~# drbd-overview

0:llhd1r02n05/0 Connected Primary/Secondary UpToDate/UpToDate /pitrix/data/container xfs 4.1T 3.6T 508G 88%

1:llhd1r01n05/0 StandAlone Primary/Secondary UpToDate/DUnknown

root@llhd1r02n05:~# drbdadm secondary llhd1r01n05

1: State change failed: (-12) Device is held open by someone

Command 'drbdsetup-84 secondary 1' terminated with exit code 11

先查下什么进程占用了盘

root@llhd1r02n05:/proc# grep drbd1 \*/mounts

8499/mounts:/dev/drbd1 /pitrix/data/container\_s xfs rw,noatime,attr2,inode64,allocsize=16384k,logbsize=256k,sunit=512,swidth=5120,noquota 0 0

r

查看进程 8499 是什么

root@llhd1r02n05:/proc# ps -ef | grep 8499

dhcpd 8499 1 0 Mar12 ? 00:00:00 /usr/sbin/dhcpd -user dhcpd -group dhcpd --no-pid -f -q -4 -cf /pitrix/dvr/rtr-jpx80vph\_dhcpd.cfg -lf /pitrix/dvr/rtr-jpx80vph\_dhcpd.leases kr\_esac095\_p

重启这个服务

service rtr-jpx80vph\_dhcpd restart

这样 drbd1 的占用就会被释放掉

#### Diskless 状态的处理

示例

root@cms1r01n02:~#drbd-overview

0:cms1r01n02/0 Connected Primary/Secondary UpToDate/Diskless C r----- /pitrix/data/container xfs 19T 205G 18T 2%

1:cms1r01n03/0 Connected Secondary/Primary UpToDate/Diskless C r-----

diskless原因通常是磁盘故障/raid卡故障，或者是操作系统kernel panic引起的，任何操作都会hang，所以无法远程重启，必须现场手工重启硬件。

#### 发生灾难迁移之后的处理

发生灾难迁移后，备机上的备盘被挂在到/pitrix/data/container\_s

root@llhd1r01n05:~# drbd-overview

0:llhd1r01n05/0 WFConnection Primary/Unknown UpToDate/DUnknown /pitrix/data/container xfs 4.1T 3.7T 467G 89%

1:llhd1r02n05/0 StandAlone Primary/Unknown UpToDate/DUnknown /pitrix/data/container\_s xfs 4.1T 3.6T 508G 88%

等待迁移任务结束后，通常能够container\_s会自动卸载掉，启动宕机的机器，系统会自动开始同步 但可能会出现container\_s未被卸载的情况，这时检查container\_s目录下是否还有文件没有被迁移走(archive除外)， 如果已迁移走，可以手工umount containers，之后系统自动开始同步修复 如果能够确认迁移完成，但是文件还在，可以把文件mv到archive中，再umount，请谨慎确认是否迁移完成，迁移结果是否正确。

umount /pitrix/data/container\_s

如果container\_s卸载失败，一般有程序占用，参考5中的方法以及 lsof /pitrix/data/container\_s查询， 先解决占用问题

#### 通用drbd恢复方法

##### 使用Installer提供的脚本修复

首先，将需要修复的2个节点上的资源全部迁移走。

然后，将rc.local中有关supervisord自启动的命令注释掉。

最后，重启节点，观察drbd的状态，若状态一致则按下方步骤继续进行。

# 确保 devops1ar01n03 节点的 drbd 的状态如下

root@devops1ar01n03:~# drbd-overview

0:devops1ar01n03/0 Unconfigured . .

1:devops1ar01n04/0 Unconfigured . .

# 确保 devops1ar01n04 节点的 drbd 的状态如下

root@devops1ar01n04:~# drbd-overview

0:devops1ar01n04/0 Unconfigured . .

1:devops1ar01n03/0 Unconfigured . .

# 查看脚本的帮助信息

root@qingcloud-firstbox:~# /pitrix/bin/repair\_drbd.sh -h

# 恢复一组 drbd 资源

root@qingcloud-firstbox:~# /pitrix/bin/repair\_drbd.sh -p devops1ar01n03 -s devops1ar01n04

# 正确的返回的结果

root@devops1ar01n03:~# drbd-overview

0:devops1ar01n03/0 Connected Primary/Secondary UpToDate/UpToDate /pitrix/data/container ext4 99G 61M 99G 1%

1:devops1ar01n04/0 Unconfigured . .

root@devops1ar01n04:~# drbd-overview

0:devops1ar01n04/0 Unconfigured . .

1:devops1ar01n03/0 Connected Secondary/Primary UpToDate/UpToDate

# 恢复一组 drbd 资源

root@qingcloud-firstbox:~# /pitrix/bin/repair\_drbd.sh -p devops1ar01n04 -s devops1ar01n03

# 正确的返回的结果

root@devops1ar01n03:~# drbd-overview

0:devops1ar01n03/0 Connected Primary/Secondary UpToDate/UpToDate /pitrix/data/container ext4 99G 61M 99G 1%

1:devops1ar01n04/0 Connected Secondary/Primary UpToDate/UpToDate

root@devops1ar01n04:~# drbd-overview

0:devops1ar01n04/0 Connected Primary/Secondary UpToDate/UpToDate /pitrix/data/container ext4 99G 61M 99G 1%

1:devops1ar01n03/0 Connected Secondary/Primary UpToDate/UpToDate

# 将 rc.local 中 supervisord 的自启动打开， 然后重启服务器，观察drbd的状态是否正常

##### 手动修复的方法

假设A节点 -> B节点同步，A节点为primary，B节点为secondary

# A节点的当前状态

0:A/0 Connected Unconfigured/Unconfigured UpToDate/UpToDate

# B节点的当前状态

0:A/0 Connected Unconfigured/Unconfigured UpToDate/UpToDate

# 将2个节点的`compute\_agent`服务停止

supervisorctl stop compute\_agent

# 将2个节点的垃圾信息删除掉

rm -f /etc/drbd.d/{\*.init\*,\*.rebuild}

# 清除磁盘的脏数据，按照实际情况修改 of 后的磁盘

dd if=/dev/zero of=/dev/intelcas1-1p1 bs=1M count=128

# A节点，执行如下操作

umount /pitrix/data/container

drbdadm down A

# B节点，执行如下操作

drbdadm down A

drbdadm -- --force create-md A

drbdadm up A

# A节点，执行如下操作

drbdadm down A

drbdadm -- --force create-md A

drbdadm up A

drbdadm -- --clear-bitmap new-current-uuid A

drbdadm primary --force A/0

## 通过如下命令，确认磁盘格式，若存在该文件，则为ext4格式

ll /etc/qingcloud/drbd\_with\_ext4

## 若为ext4，请使用如下命令

mkfs.ext4 -q -E nodiscard,lazy\_itable\_init=0,lazy\_journal\_init=0 -O sparse\_super,large\_file -m 0 -i 67108864 /dev/drbd0

mount -t ext4 -o defaults,noatime /dev/drbd0 /pitrix/data/container

## 若非ext4，请使用如下命令(老环境)

mkfs.xfs -f -K /dev/drbd0

mount -t xfs -o defaults,allocsize=16m,noatime,nobarrier /dev/drbd0 /pitrix/data/container

# A节点的正确状态

0:A/0 Connected Primary/Secondary UpToDate/UpToDate /pitrix/data/container xfs 1.5T 33G 1.5T 3%

# B节点的正确状态

0:A/0 Connected Secondary/Primary UpToDate/UpToDate

# 将2个节点的`compute\_agent`服务启动

supervisorctl start compute\_agent

##### rebuild方法

1）将两边的`compute\_agent`停掉，`supervisorctl stop compute\_agent`。

2）在两边确保`/pitrix/data/container`没有`mount`，如果已经加载，运行 `umount /pitrix/data/container`。

3）运行 `drbdadm down xxx` 将`xxx`节点`drbd`停掉。

4）再去 `touch /etc/drbd.d/xxx.rebuild`，然后重启`compute\_agent`就会正常做`rebuild`。

### ZFS故障修复

#### 如何查看zfs状态

可以通过zfs status 命令查看本机zfs各盘状态，命令输出结果示例如下

root@ztgq2r02n04:~# zpool status

pool: vpool

state: ONLINE

scan: none requested

config:

NAME STATE READ WRITE CKSUM

vpool ONLINE 0 0 0

nvme0n1 ONLINE 0 0 0

nvme1n1 ONLINE 0 0 0

errors: No known data errors

#### ZFS换盘步骤

1）使用 zpool offline 命令使磁盘脱机（如有必要）。

2）移除要替换的磁盘。

3）插入替换磁盘。

4）运行 zpool replace 命令将新盘替换旧盘的id，例如：

zpool replace -f vpool sda sdx

# 如果是flash卡

zpool create -f -o ashift=12 vpool /dev/nvme0n1

zpool create -f -o ashift=12 vpool /dev/nvme0n1 /dev/nvme1n1

注：如果用by-id方法创建vpool，换盘也需要使用磁盘wwn进行操作

5）使用 zpool online 命令使磁盘联机。

#### ZFS重建步骤

如果vpool里的ssd盘拔出，或卡损坏，造成了vpool不可用，则需要重新构建vpool。重建步骤如下：

1. supervisorctl stop compute\_agent
2. 重启服务器；
3. 确认zfs驱动正常加载，若未加载需要执行

/sbin/modprobe zfs

1. 重建vpool

zpool create vpool -o ashift=12 -f /dev/nvme0n1 /dev/nvme1n1

## 青云目录结构

青云部署在每台服务器的/pitrix目录下，在/pitrix目录下一般会有存在如下子目录，目录说明如下：

bin 放可执行文件和一些辅助脚本

cli 命令行

conf 进程配置文件

data/container 用户数据

data/image 模板数据

lock 锁文件

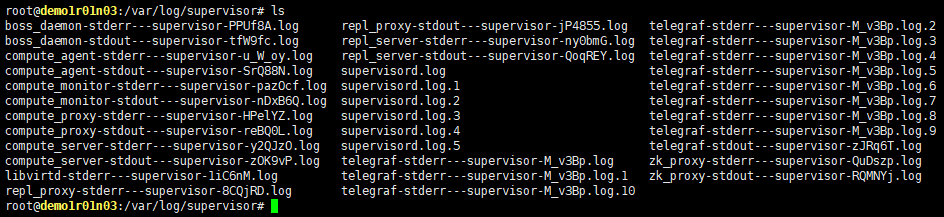
log 每个进程日志文件

notifier 告警文件推送目录

## 青云系统日志

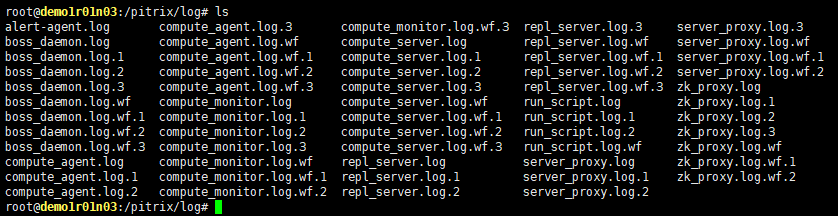
### Supervisor服务说明

青云的服务模块统一由 supervisor 管理。在管理节点、计算节点、存储节点上都有supervisor 进程运行。supervisor 会自动尝试恢复服务。可登陆到设备， 用 supervisorctl status 观察服务状态，若尚未被恢复，可 supervisorctl restart 来手动恢复某服务。然后再寻找服务异常的原因。 supervisor 的log都按模块分别放在 /var/log/supervisor/ 目录下，可找到相应 log 并分析原因。



### 青云服务模块日志

在管理节点、计算节点、存储节点上都有青云的服务模块运行。log的位置都按模块放在该节点的 /pitrix/log 目录下，wf结尾的为相应服务的报错LOG。



### 其他重要日志

Linux一些重要的系统log有时也需要查看。这些 log 默认都放在 /var/log 目录下，例如 kern.log, syslog, messages, dmesg 等。

## 日常巡检

### 平台状态巡检

平台状态巡检主要是通过firstbox检查IAAS平台内的一些关键信息，其他详细巡检内容可参考《IAAS平台巡检手册》。

#### 检查所有节点时间是否一致

执行命令如下：

root@qingcloud-firstbox:~# /pitrix/upgrade/exec\_nodes.sh all 'date'

返回结果示例：

Execing [demo1r01n00] with [date] ...

Fri Feb 7 10:36:18 CST 2020

Execing [demo1r01n03] with [date] ...

Fri Feb 7 10:36:18 CST 2020

Execing [demo1r01n01] with [date] ...

Fri Feb 7 10:36:18 CST 2020

#### 检查所有节点NTP服务状态

执行命令如下：

root@qingcloud-firstbox:~# /pitrix/upgrade/exec\_nodes.sh all 'ntpq -np'

返回结果示例（环境中的一到两个节点会指向平台外部的ntp，其余节点均指向平台内部的NTPServer）：

Execing [demo1r01n00] with [ntpq -np] ...

remote refid st t when poll reach delay offset jitter

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

+144.76.76.107 205.46.178.169 2 u 1517 1024 356 182.746 -9.740 2.254

\*84.16.67.12 .GPS. 1 u 1643 1024 202 173.935 -10.107 5.967

108.59.2.24 130.133.1.10 2 u 277m 1024 0 223.936 0.521 0.000

-193.182.111.143 194.58.202.20 2 u 407 1024 367 205.300 3.851 2.146

+91.189.89.198 145.238.203.14 2 u 630 1024 377 187.543 -15.243 16.567

Execing [demo1r01n02] with [ntpq -np] ...

remote refid st t when poll reach delay offset jitter

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

\*200.1.1.10 84.16.67.12 2 u 1051 1024 377 0.075 -0.658 7.974

+200.1.1.20 139.199.214.202 3 u 716 1024 377 0.067 -2.806 3.213

Execing [pgpool1] with [ntpq -np] ...

remote refid st t when poll reach delay offset jitter

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

\*200.1.1.10 84.16.67.12 2 u 343 1024 377 0.255 0.091 4.711

+200.1.1.20 139.199.214.202 3 u 976 1024 377 0.282 -2.227 1.477

#### 检查所有节点守护进程服务状态

执行命令如下：

root@qingcloud-firstbox:~# /pitrix/upgrade/exec\_nodes.sh all 'supervisorctl status'

返回结果示例：

Execing [demo1r01n01] with [supervisorctl status] ...

libvirtd RUNNING pid 1978, uptime 2 days, 9:00:38

storage\_proxy RUNNING pid 1979, uptime 2 days, 9:00:38

storage\_server RUNNING pid 14783, uptime 1 day, 23:55:56

storage\_vbr RUNNING pid 14782, uptime 1 day, 23:55:56

storage\_vbr\_proxy RUNNING pid 1976, uptime 2 days, 9:00:38

telegraf RUNNING pid 1973, uptime 2 days, 9:00:38

zk\_proxy RUNNING pid 1974, uptime 2 days, 9:00:38

Execing [dns-slave] with [supervisorctl status] ...

metadata\_server RUNNING pid 934, uptime 3 days, 20:39:50

name\_agent RUNNING pid 935, uptime 3 days, 20:39:50

name\_proxy RUNNING pid 936, uptime 3 days, 20:39:50

name\_server RUNNING pid 6348, uptime 2 days, 22:52:05

telegraf RUNNING pid 933, uptime 3 days, 20:39:50

#### 检查所有节点存储空间

执行命令如下：

root@qingcloud-firstbox:~# /pitrix/upgrade/exec\_nodes.sh all ' df -h|grep dev'

返回结果示例：

Execing [demo1r01n00] with [df -h|grep dev] ...

udev 63G 8.0K 63G 1% /dev

/dev/sda1 46G 41G 3.2G 93% /

/dev/sda3 482G 467G 16G 97% /pitrix

/dev/sdb1 9.0T 1.5T 7.6T 16% /home/installer

/dev/nbd14p1 20G 1.7G 17G 10% /mnt

Execing [demo1r01n01] with [df -h|grep dev] ...

/dev/sda1 46G 35G 8.9G 80% /

udev 32G 4.0K 32G 1% /dev

/dev/sdb1 9.0T 3.0T 6.1T 33% /pitrix

#### 检查所有hyper节点磁盘状态

执行命令如下：

root@qingcloud-firstbox:~# /pitrix/upgrade/exec\_nodes.sh hyper-pair 'drbd-overview'

root@qingcloud-firstbox:~# /pitrix/upgrade/exec\_nodes.sh hyper-repl ' zpool status'

返回结果示例：

hyper-pair

Execing [demo1r01n04] with [drbd-overview] ...

0:demo1r01n04/0 Connected Primary/Secondary UpToDate/UpToDate /pitrix/data/container xfs 4.1T 813G 3.3T 20%

1:demo1r02n04/0 Connected Secondary/Primary UpToDate/UpToDate

Execing [demo1r02n03] with [drbd-overview] ...

0:demo1r02n03/0 Connected Primary/Secondary UpToDate/UpToDate /pitrix/data/container xfs 4.1T 1.1T 3.1T 27%

1:demo1r01n03/0 Connected Secondary/Primary UpToDate/UpToDate

hyper-repl

Execing [ztgq2r01n04] with [zpool status] ...

pool: vpool

state: ONLINE

scan: none requested

config:

NAME STATE READ WRITE CKSUM

vpool ONLINE 0 0 0

nvme0n1 ONLINE 0 0 0

nvme1n1 ONLINE 0 0 0

errors: No known data errors

Execing [ztgq2r02n05] with [zpool status] ...

pool: vpool

state: ONLINE

scan: none requested

config:

NAME STATE READ WRITE CKSUM

vpool ONLINE 0 0 0

nvme0n1 ONLINE 0 0 0

nvme1n1 ONLINE 0 0 0

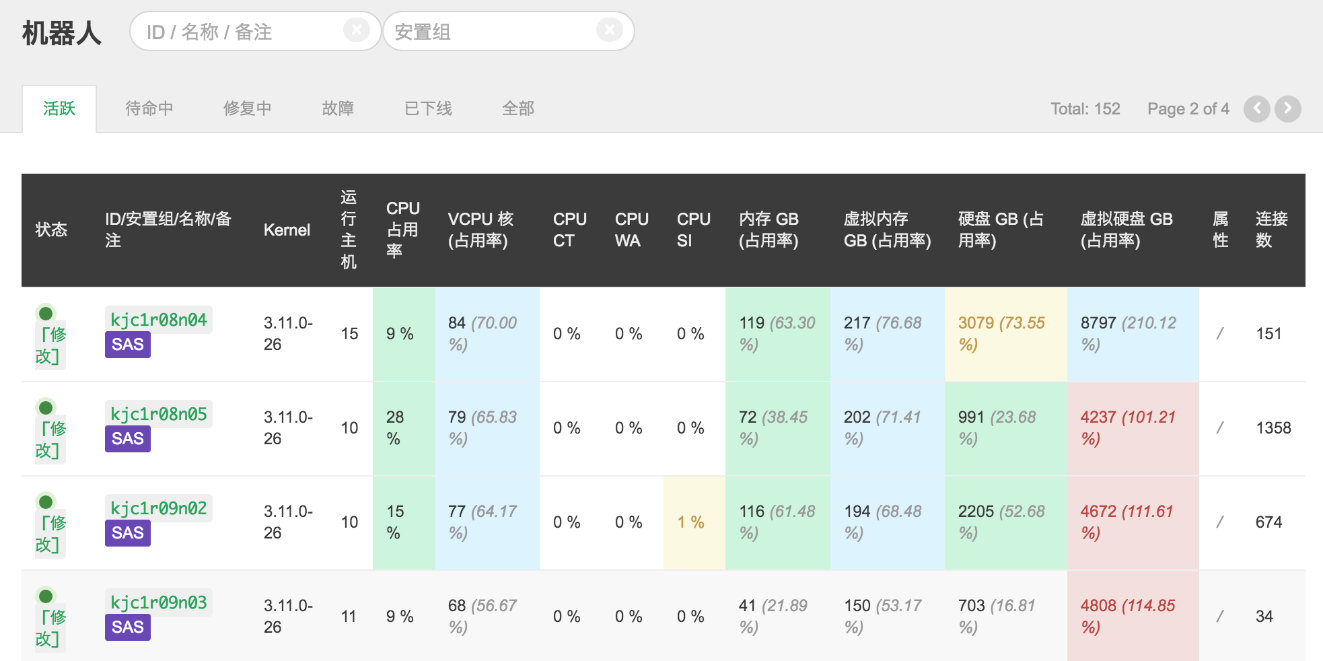
errors: No known data errors

### 资源容量监控

目前对资源容量进行监控最直接的手段是通过supervisor管理控制台，比如监控物理服务器资源使用情况和EIP池使用情况。

#### 物理服务器监控

Supervisor控制台🡪选择要查看的zone🡪机器人



##### 服务器状态

物理服务器的分为6种，活跃、待命中、修复中、故障、已下线、全部，服务器的状态可以人为修改，修改后，服务器即进入对应的状态类别列表里，服务器宕机或人为关机后，服务器状态也会自动改变。

|  |  |
| --- | --- |
| 资源状态 | 说明 |
| 活跃 | 正常运行中的服务器，在创建资源时、迁移资源时会将这些主机做为备选目标 |
| 待命中 | 当有服务器宕机，其pair节点的状态会自动变为待命中，新创建的资源及迁移资源时，将不会把这类主机做为备选目标，有服务器宕机时，应及时维修，维修完成后将这组pair的主机状态改为active；  某些情况下需要将某些物理主机给特定的资源使用，这时可以将这台主机的状态人为设置为待命中，平台将不会自动向这台主机分配资源（例外情况：当pair的节点一台处于活跃，一台处于待命中，其中任何一台主机宕机，不影响在另外一台上恢复资源。），但可以通过人工迁移方式指定将资源迁移到待命中的某台节点；  当服务器出现硬件报警时，需要将其上的资源迁移到其它pair，并把这个pair的服务器状态置为待命中。 |
| 修复中 | 如果服务器正在维修，可以把状态改为修复中，这个好像一般用不到。服务器故障后，一般状态会自动变为故障，维修后决定将机器状态改为活跃或待命中。 |
| 故障 | 服务器异常宕机或人为关机后，状态会自动改为故障，需尽快维修并根据需要设置为活跃或待命中，如果短时间内无法恢复，需要将pair上的资源迁移到健康的pair；  当服务器处于故障状态时，将不会检查其pair节点的状态。 |
| 已下线 | 不再使用的服务器，也可以将这类服务器从数据库中清除。 |
| 全部 | 列出所有状态的物理服务器。 |

##### 资源监控说明

通过查看服务器的实时资源使用情况，判断当前服务器的压力及瓶颈，有的时候我们需要在某些物理机使用量过高时做rebalance。

例如物理机的CPU/内存/CT(说明IO严重不足)/SI 太高时，都需要进行rebalance，确保物理机压力尽量均等。（其实我们bot的placement策略已经有考虑，但是资源确实有突发使用的情况）

|  |  |  |
| --- | --- | --- |
| 字段类型 | 字段 | 说明 |
| 基本信息 | 状态 | 参考服务器状态章节 |
| ID/安置组/名称/备注 | 显示主机名及主机所在的资源组，目前资源组包含：SAS、SSD、SATA，代表主机的类型，这个类型是在安装Hyper时候定义的。 |
| Kernel | 宿主机内核版本信息 |
| 运行主机 | 这台主机上正在运行的vm数量，不包括已关机的vm。 |
| 物理资源 | CPU 占用率 | 物理机实际CPU使用率，包括物理机自己占用的和虚拟资源占用的。 |
| 内存 GB (占用率) | 物理机实际MEM使用率，包括物理机自己占用的和虚拟资源占用的。 |
| 硬盘 GB (占用率) | 物理机实际存储空间使用率。  实际存储容量计算方法：  每台计算节点12块900G的sas盘，做raid后，容量为9000G，其中4500G用于生产数据，另外4500G用于存储数据副本，即实际可用空间约为4500G，约4.39T，因为生产厂商和计算机对磁盘容量计算差异的原因，实际看到的磁盘空间为磁盘厂商标注容量的约93%，4.39T x 93%＝ 4.08T，约为4.1T |
| 逻辑资源 | VCPU 核 (占用率) | 物理机逻辑CPU使用率，这个数据抓取的是运行在这台物理机上每台VM最高CPU的配置值（创建vm时，在选择配置步骤，可以看到最大CPU参数项，这个值表示此VM可以在线扩容到的CPU上限。），  CPU通常会超卖，一般超卖3倍或5倍，计算方法如下：每台计算节服务器2颗物理CPU，每颗CPU 8 core 16线程，2颗CPU共32线程，系统设置CPU超卖3倍，32 x 3 =96超卖比的设置用server.yaml中的cpu\_oversale\_rate参数 |
| 虚拟内存 GB (占用率) | 物理机逻辑MEM使用率。这个数据抓取的是运行在这台物理机上每台VM最高MEM的配置值（创建vm时，在选择配置步骤，可以看到最大内存参数项，这个值表示此VM可以在线扩容到的内存上限。），虚拟内存一般不超卖，虚拟内存大小一般等于物理内存大小。 |
| 虚拟硬盘 GB (占用率) | 物理机逻辑存储空间使用率，一般超卖设置为0.2，代表超卖5倍。  超卖比的设置用server.yaml中的disk\_reserve\_rate参数 |
| 服务器性能 | CPU CT | 可以说明服务器IO压力，这个值如果不是0说明服务器IO有一定压力，值越大说明IO压力越大，甚至是说明IO严重不足，此时通过iostat -x 1可以查看utils应该非常高。可以通过iotop找到io占用高的vm并迁移。  Utils说明：一秒中有百分之多少的时间用于 I/O 操作，或者说一秒中有多少时间 I/O 队列是非空的。如果 %util 接近 100%，说明产生的I/O请求太多，I/O系统已经满负荷，该磁盘可能存在瓶颈。 |
| CPU WA | 如果是非0表示已经有产生了io wait，即io等待，同时会有io争抢，此时看CPU CT应该是比较高的，因为IO比较繁忙，所以产生了io wait，此值越大，说明io wait越严重，会导致服务器性能下降，需要进行迁移，如果是持续对IO压力大，需考虑迁移到SSD节点。 |
| CPU SI | CPU软中断，一般都是网络太忙导致，例如有高并发的vm等。SI太高会占用CPU性能，同时还会影响IO性能（因为我们需要通过网络进行同步），甚至还会影响迁 移速度（因为迁移bbcp也依赖网络）。这时候需要迁移网络压力大的vm。 |
| 其它 | 属性 | 暂时没有用。 |
| 连接数 | 通过/usr/sbin/conntrack -C命令得到的数值，从/pitrix/bin/alert\_agent中查看，这个值小于10000是安全的。  可以通过/usr/sbin/conntrack –L命令列出当前服务器所有的连接。  百度百科对conntrack的解释：  连接跟踪（CONNTRACK），顾名思义，就是跟踪并且记录连接状态。Linux为每一个经过网络堆栈的数据包，生成一个新的连接记录项 （Connection entry）。 |

#### 公网IP池

Supervisor控制台🡪选择要查看的zone🡪公网IP池

在公网IP池列表中，点击公网IP分组边上的下三角🡪全部，会显示全部EIP组中的IP



##### EIP状态

|  |  |
| --- | --- |
| 资源状态 | 说明 |
| 已占用 | 已被申请出去的EIP。 |
| 可用 | 可被申请的EIP。 |
| 已禁用 | 处于禁用状态的IP，需启用后才可以被申请到，新添加进来的EIP默认状态。 |
| 全部 | 显示全部EIP地址 |

##### EIP资源容量

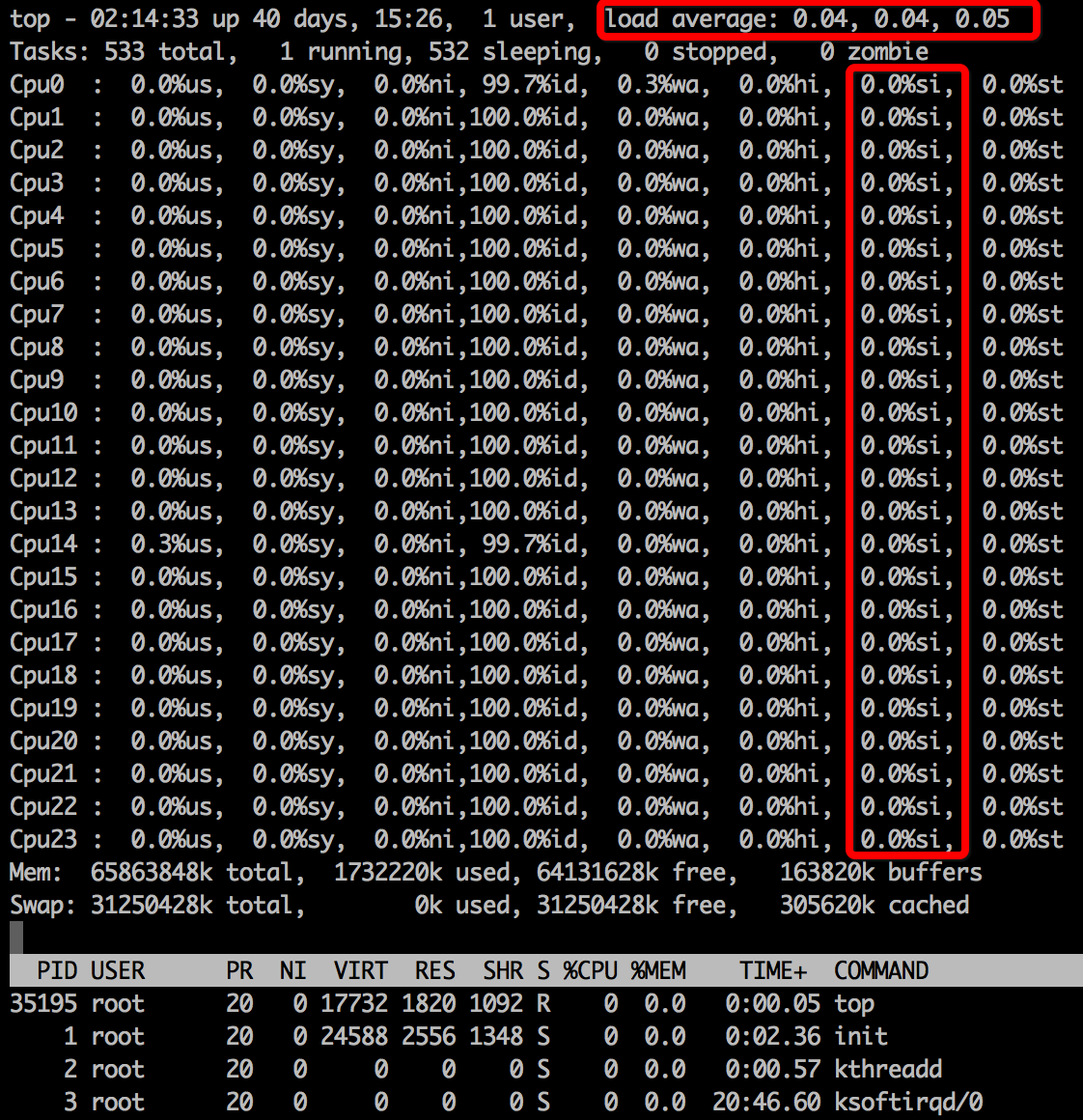
当可用EIP不足时，需要及时添加，以免业务申请不到新的EIP地址。

##### VG服务器绑定EIP数量

目前我们建议每台VG最多配置绑定1个C的IP地址，超过1个C后，需要考虑扩容VG。

通过iptables -S查看当前vg绑定的EIP。

在实际环境中，VG中绑定的EIP数量也要看VG服务器的负载，打开top工具，按1显示所有cpu核。



|  |  |
| --- | --- |
| 参数 | 说明 |
| Load average | load average的三个值分别代表过去一分钟、五分钟、十五分钟内CPU的负载情况，这个值超过1代表压力就有些大了，关于load average的说明 |
| Si | 代表每个核的软中断处理负载，数值越大说明这个cpu核越忙，一般超过15说明这个CPU核的软件中断处理压力比较大，超过15时会有报警邮件，这个值与cpu的型号有直接关系，配置较好的CPU软中断处理能力也高。  如果单个核的si比较高，说明可能有某个IP的流量非常大，估计进出双方向的流量合计在400Mbps以上，需要确认流量为正常流量，如果是正常流量需要根据业务进行流量拆分，或更换更高配置CPU的VG服务器。 |

#### 扩容考虑依据

我目前是每天通过机器人界面看一下资源使用情况，一般CPU、MEM资源使用率到50%-60%左右，提醒用户准备扩容，或告诉客户暂时不能再申请资源。

资源使用率我目前有下面几个考虑，供参考。

1）除了SATA的节点，存储资源一般不会成为瓶颈，SATA需要保证一个PAIR宕掉后，其它的节点可以保证资源的迁移，并有30%左右的剩余空间。

简单的算法是：假设任何一个PAIR宕掉，将上面的已存的数据容量加到其它PAIR中，然后其它PAIR的总容量仍然剩余30%左右。

2）CPU和MEM，物理使用率和虚拟使用率使用上线大概在60%左右，在一半左右的机器使用率在50%-60%左右时，考虑扩容，不过利用mig5的功能，在具有一定规模的环境，这个使用率可能再高一些，对灾难迁移应该也没有影响了，不过还是不能太高，太高vm性能也会下降，还是以保证我们自己安全为主。

3）VG，每台vg上绑定的EIP数量大概在1个C，当然也要考虑每个EIP流量的负载，一般是超过一个C后增加新的VG节点。

4）可能某台物理机上的CPU、MEM、DISK使用率会比较高，需要进行人工balance。

5）如果普遍IO使用率高（SAS节点），需要扩容，或增加SSD节点，偶尔的磁盘使用率高或持续的某台IO使用率高，可以考虑balance

6）服务器的CPU CT、CPU WA、CPU SI比较高，需要考虑rebalance服务器上的资源。

## 迁移虚拟机

### 通过supervisor迁移

1）如果是由于物理机有问题需要迁移资源，先将故障的物理机状态设置为“待命中”，避免再有新的资源分配到这台物理机。

supervisor左侧导航—>机器人，点击故障机器状态栏字段的修改，将状态改为“待命中(standby)”，备注里写上操作原因，点击“修改”。

2）查看这台主机上运行了哪些vm

supervisor左侧导航—>主机，在右边内容显示区最上面的“物理机”搜索框，搜索故障主机的hostname（如：jzsec2r02n05），会显示这台主机上运行的vm。

3）迁移vm

supervisor左侧导航—>主机，在右边内容显示区最上面的“ID/名称/用户”搜索框，搜索对应的VM ID，点开此主机—>迁移主机—>页面下方的“手动选择物理机”或“系统指定物理机”，建议选择“系统指定物理机”，如果手动选择，需要在列表中选择一台负载较低的物理机。





### 使用cli迁移

1）登录webservice0或webservice1，进入cli目录

# cd /pitrix/cli/

2）使用migrage-instances迁移

./migrate-instances –i instance-id -t TARGET\_HOST

-i INSTANCES, --instances=INSTANCES

-t TARGET\_HOST, --target\_host=TARGET\_HOST

## 迁移hyper

资源的迁移，可以指定一次迁移一个资源，也可以指定一次迁移某几个资源，也可以迁移整个Hyper。

1）登录webservice0或webservice1，进入cli目录

# cd /pitrix/cli/

2）指定将Hyper上的资源迁移到某台目标Hyper

# ./migrate-hyper-node -H source\_hyper\_node -t target\_hyper\_node

-H 表示被迁移的Hyper

-t 表示迁移到的目标Hyper，如果不指定此参数，系统将根据资源池负载为资源自动选择目标Hyper。

## VG节点迁移

### 单个IP手动迁移

1）列出要被迁移VG上所有的EIP，在VG上运行iptables -S

root@kjc1r07n00:~#iptables -S

-P INPUT ACCEPT

-P FORWARD ACCEPT

-P OUTPUT ACCEPT

-A FORWARD -s 10.17.0.0/16 -j ACCEPT

-A FORWARD -d 10.17.0.0/16 -j ACCEPT

-A FORWARD -s 10.54.159.28/32 -m comment --comment "114.113.64.16" -j ACCEPT

-A FORWARD -d 10.54.159.28/32 -m comment --comment "114.113.64.16" -j ACCEPT

-A FORWARD -s 10.54.35.19/32 -m comment --comment "114.113.64.10" -j ACCEPT

-A FORWARD -d 10.54.35.19/32 -m comment --comment "114.113.64.10" -j ACCEPT

按上面的输出，需要被迁移的IP地址为：

114.113.64.16

114.113.64.10

2）登录webservice0/1

# cd /pitrix/cli

3）迁移IP  
VG有主备之分，可以通过server.yaml中的定义查看角色，有HA0标识的为备，没有的为主，比如下面的114.113.64.0/27 EIP段，kjc1r07n00是主节点，kjc1r08n00是备节点。这里的主备标识只是为方便资源迁移时指定节点，两台VG为双活，都可以承载EIP，且可以互相迁移，迁移到主节点时，命令参数中不用加标识，迁移到备节点时，命令中需要加-v HA0参数。

vgateway\_kjc1r07n00:

eips:

114.113.64.0/27: '114.113.64.3(.1)'

mgmt\_ip: '10.17.21.225'

user\_ip:

10.54.0.0/16: '10.17.21.225'

vgateway\_kjc1r08n00:

eips:

114.113.64.0/27|HA0: '114.113.64.4(.1)'

mgmt\_ip: '10.17.21.227'

user\_ip:

10.54.0.0/16: '10.17.21.227'

将EIP 114.113.64.16从kjc1r07n00迁移到kjc1r08n00

# ./migrate-eips -e 114.113.64.16 -v HA0

将EIP 114.113.64.16从kjc1r08n00迁移到kjc1r07n00:

# ./migrate-eips -e 114.113.64.16

### 使用脚本批量迁移

使用migrate\_vgateway.py脚本迁移EIP。

root@webservice1:/pitrix/lib/pitrix-scripts# python migrate\_vgateway.py -h

Usage: migrate\_vgateway.py --v <vg\_mgmt\_ip> [-e <eip\_addr>] -F [-f <conf\_file>]

# only print out plan:

migrate\_vgateway.py -v 172.31.10.1 -c 10

# migrate:

migrate\_vgateway.py -v 172.31.10.1 -c 10 -F

migrate\_vgateway.py -v 172.31.10.1 -e 192.168.7.120 -F

假如我们需要迁移172.31.10.1这台vg，每次迁移10个eip，那么命令行为：

python migrate\_vgateway.py -v 172.31.10.1 -c 10 -F

假如我们需要迁移172.31.10.1这台vg，只迁移指定IP，那么命令行为：

python migrate\_vgateway.py -v 172.31.10.1 -c 10 -e 192.168.7.120 -F