H3C UIS 超融合管理平台维护手册 V2.0

目 录

1 H3C UIS 平台日常维护

为了保证局点 UIS 系统的稳定运行，需要进行维护工作。主要包括查看告警、查看操作日志、查看集群、查看主机、查看虚拟机、查看 License 以及查看日志等。

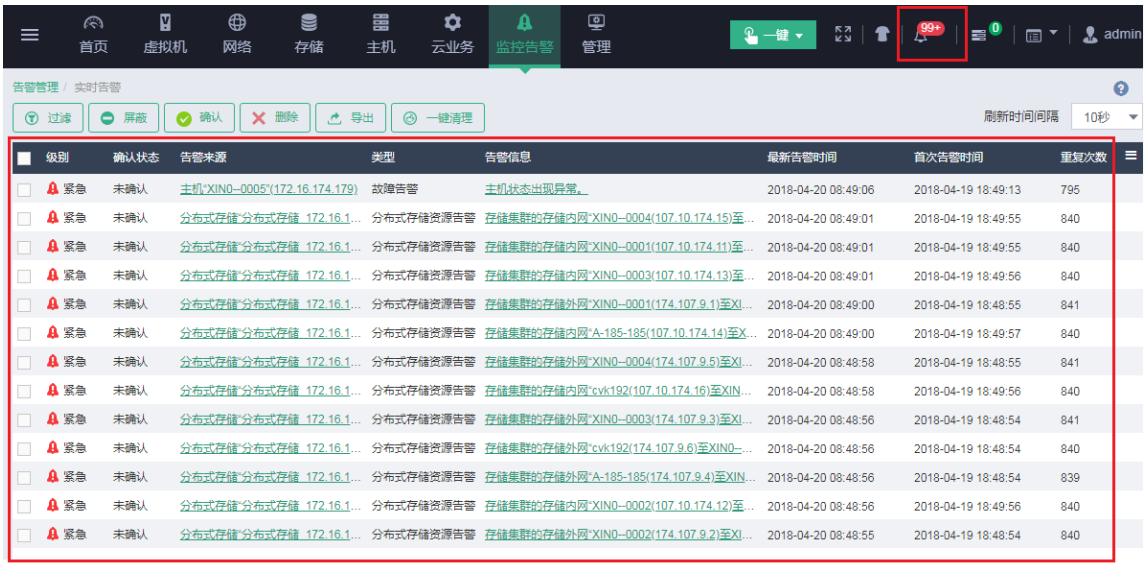
1.1 查看告警

在 UIS 平台主页面的右上方包含有 UIS 系统运行的告警指示灯，包括“紧急告警”、“重要告警”、“次要告警”和“提示告警”4 种告警指示灯。

如果“紧急告警”和“重要告警”指示灯显示有告警信息时，说明 UIS 系统运行异常，需要尽快排查问题原因并解决。

鼠标点击对应的告警指示灯，则会自动切换到“实时告警”页面，也可以通过[告警管理/实时告警]路径进入。

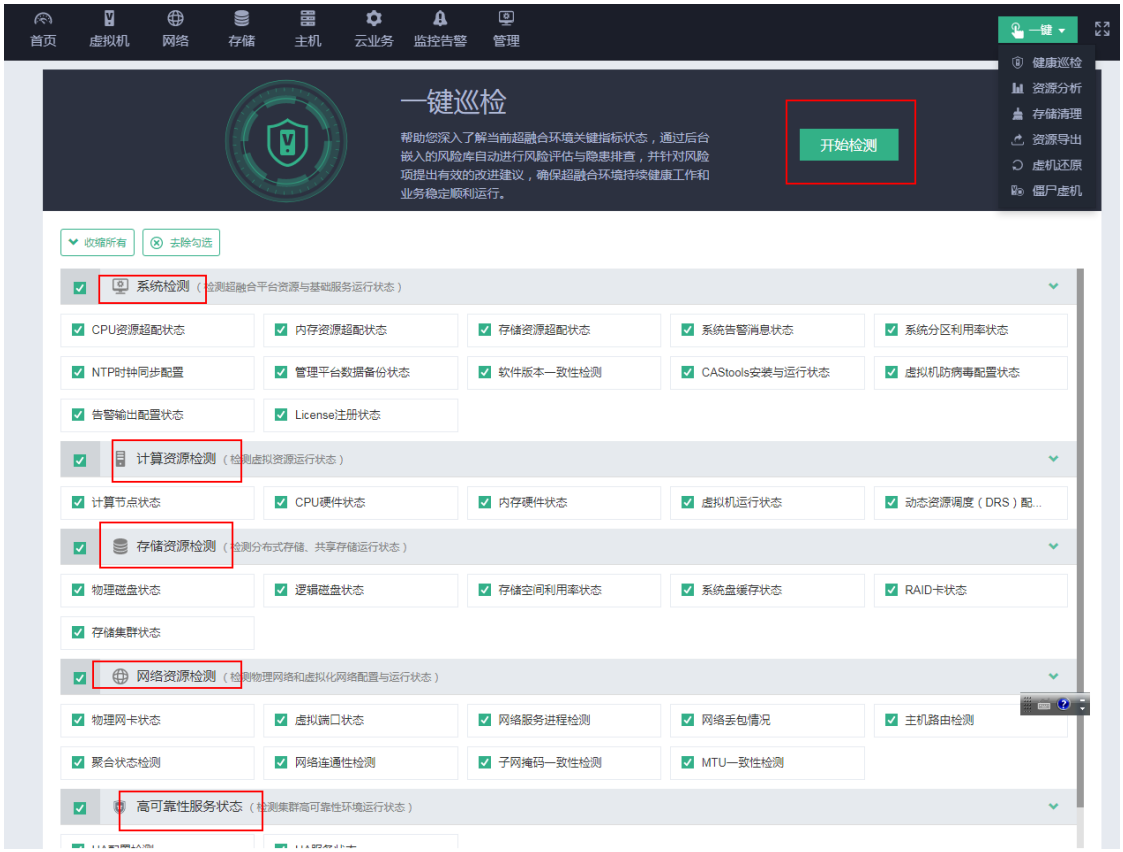
根据“实时告警”页面中的告警来源、类型、告警信息和最新告警时间进行问题排查。



1.2 一键巡检

在 UIS 平台右上方保护一键巡检功能选择。包括健康巡检、资源分析、存储清理、资源导出、虚机还原、僵尸虚机等功能。

选择“健康巡检”功能，进入已经巡检界面，可以根据需要对指定模块进行检测。



针对检测的结果，支持打印和结果导出：

如果在巡检中发现异常，例如 RAID 卡、硬盘缓存异常，可以点击修复按钮尝试修复。

1.3 查看操作日志

“操作日志”页面记录了 UIS 系统的历史操作记录信息，包括前台用户手动操作和系统后台自动操作的记录信息。

操作日志信息主要包括了操作员名称、完成时间、登录地址、操作描述、执行结果失败原因等重要信息。

如果存在失败的操作日志信息，则需要根据“失败原因”进行排查，如果操作日志太多，可下载到本地进行排查分析。

如下图是 UIS manager 的操作日志：

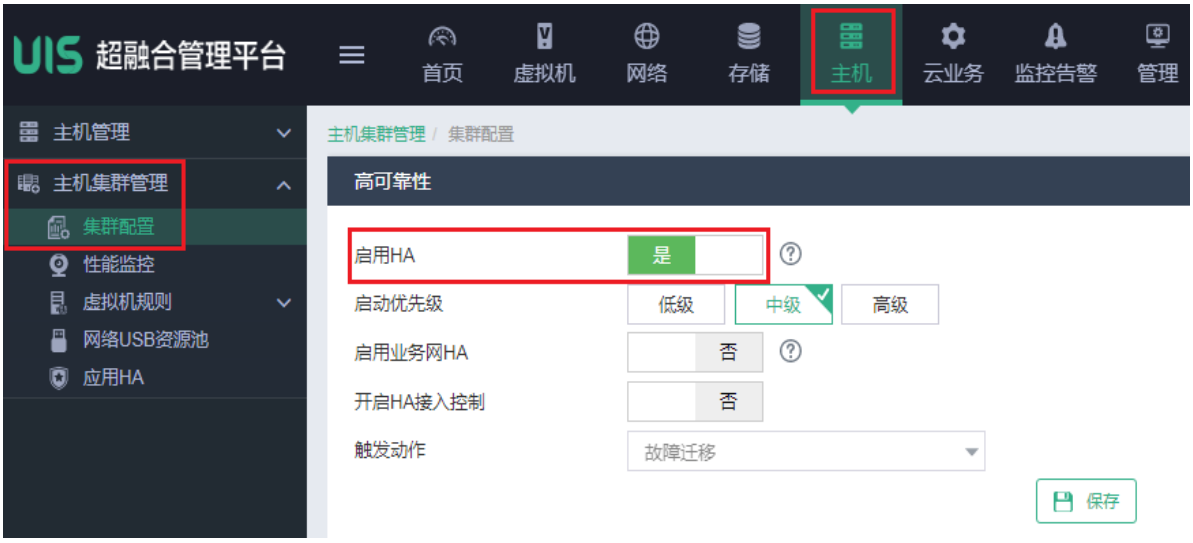
1.4 查看集群

1.4.1 查看集群的高可靠性功能

确认集群是否启用 HA 功能，如果集群没有开启 HA 功能，那么集群下的一台 CVK 主机异常时，该 CVK 上的虚拟机无法正常迁移到集群下其他的 CVK 主机。

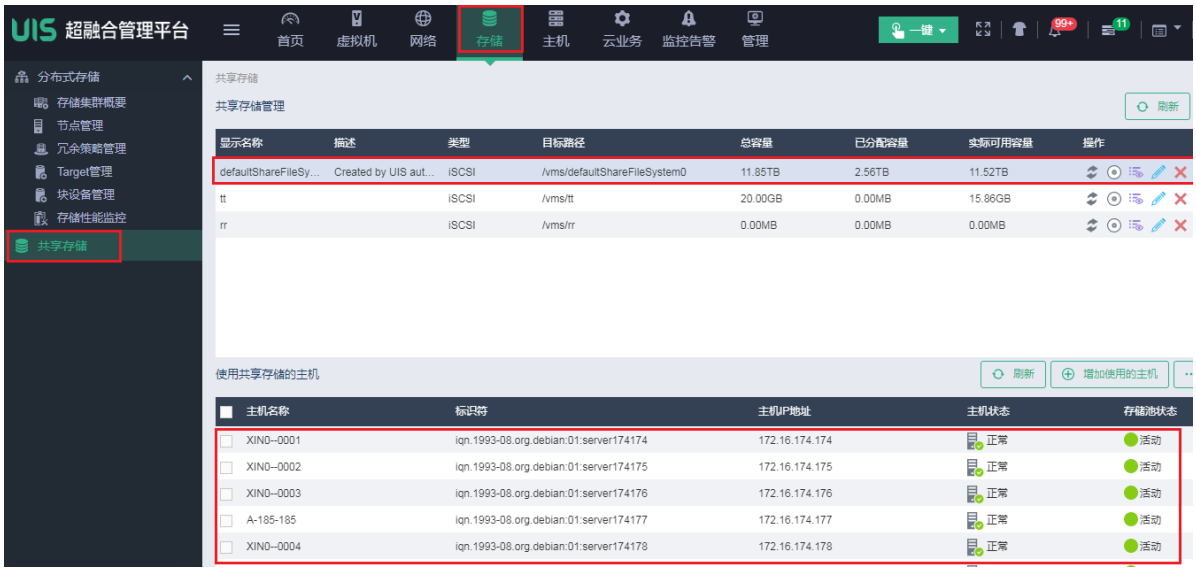
同时在已经开启 HA 的功能基础上，还可以配置启用业务区 HA，当虚拟机的业务区 HA 出现故障或者连接不通时，虚拟机可以迁移到其它主机。

启动优先级：用于设置集群中虚拟机的缺省启动优先级，包括低级、中级和高级，默认为中级。虚拟机的启动优先级在增加虚拟机或修改虚拟机的过程中设置。主机故障后，虚拟机启动的相对优先顺序。这些虚拟机在新主机上按顺序重新启动，首先启动优先级最高的虚拟机，然后是中级优先级的虚拟机，最后是低级优先级的虚拟机，直到重新启动所有虚拟机或者没有更多的可用集群资源为止。



1.4.2 查看集群下的共享存储

虚拟机发生迁移时如果目的主机没有挂载虚拟机所使用的共享存储，那么会导致虚拟机迁移失败。

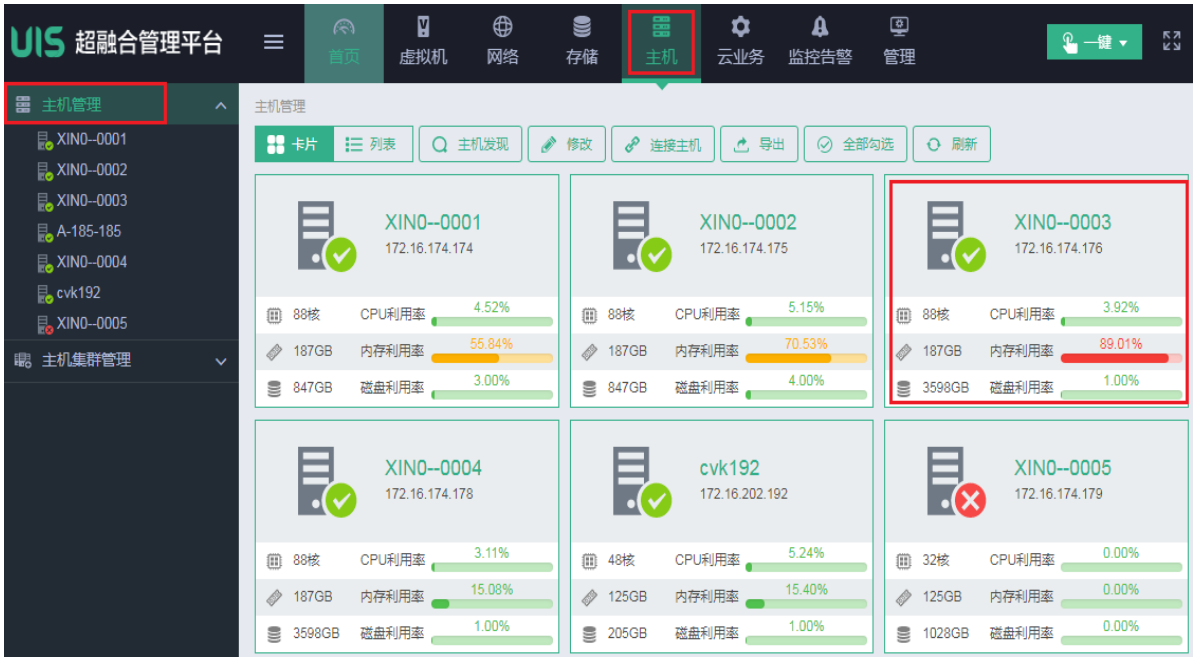


1.5 查看主机

1.5.1 查看主机状态

“查看主机”页面的主机状态，确认是否有不正常的主机。

检查各个主机的 CPU 和内存占用率是否正常，当占用率超过 80%时需要重点关注。



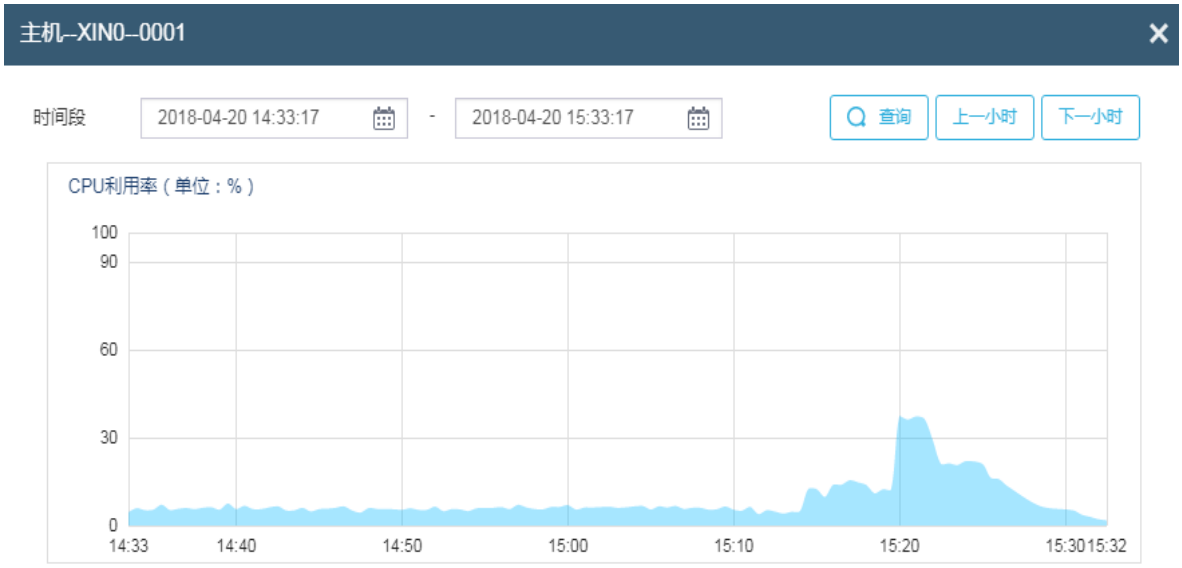
1.5.2 查看主机运行时间

在 CVK 主机的【概要】页面中可以查看到主机的详细配置信息，通过查看“运行时间”可以确定该主机近期是否有重启动作。

1.5.3 查看主机性能监控

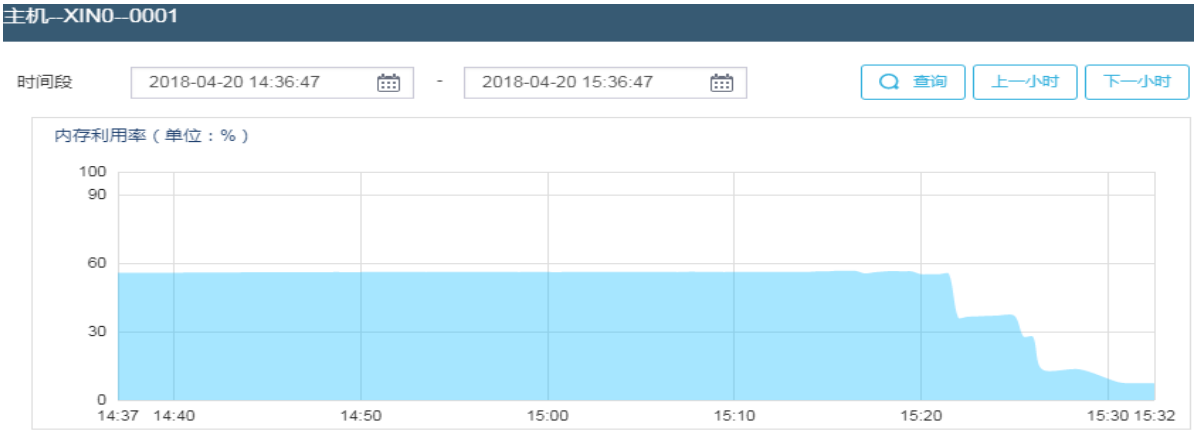
在 CVK 主机的【性能监控】页面下可以查看到主机的 CPU 利用率、内存利用率、 I/O 吞吐量、网络吞吐量、磁盘利用率和分区占用率等信息。

在[性能监控/CPU 使用情况]页面，点击<...>按钮可以查看更长时间范围内的 CPU 占用率信息。



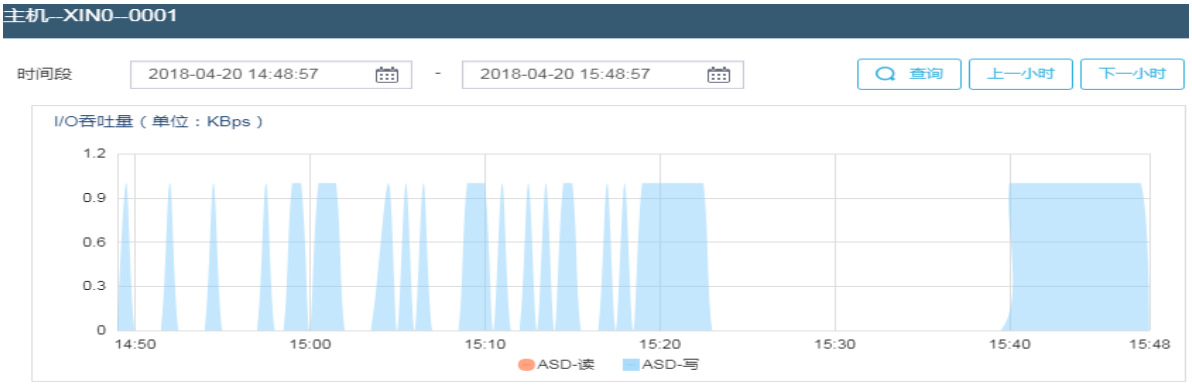
1. 查看主机内存占用率

在[性能监控/内存使用情况]页面，点击<...>按钮可以查看更长时间范围内的内存占用率信息。



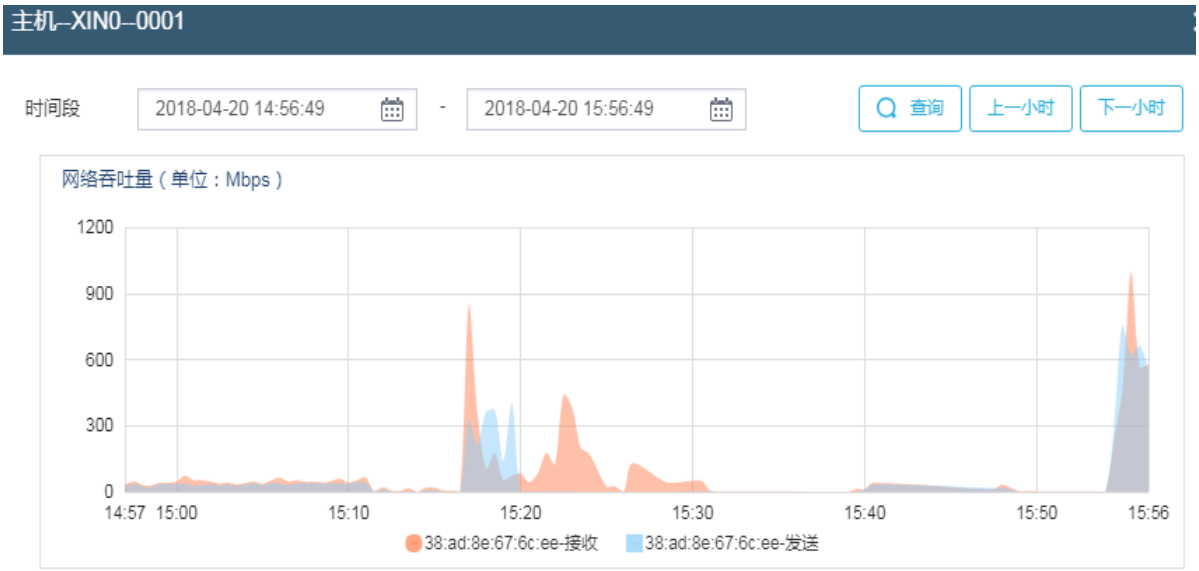
2. 查看主机 I/O 吞吐量

在[性能监控/I/O 吞吐量统计]页面可以查看主机磁盘的 I/O 吞吐量信息，点击<...>按钮可以查看更长时间范围内的 I/O 吞吐量信息。



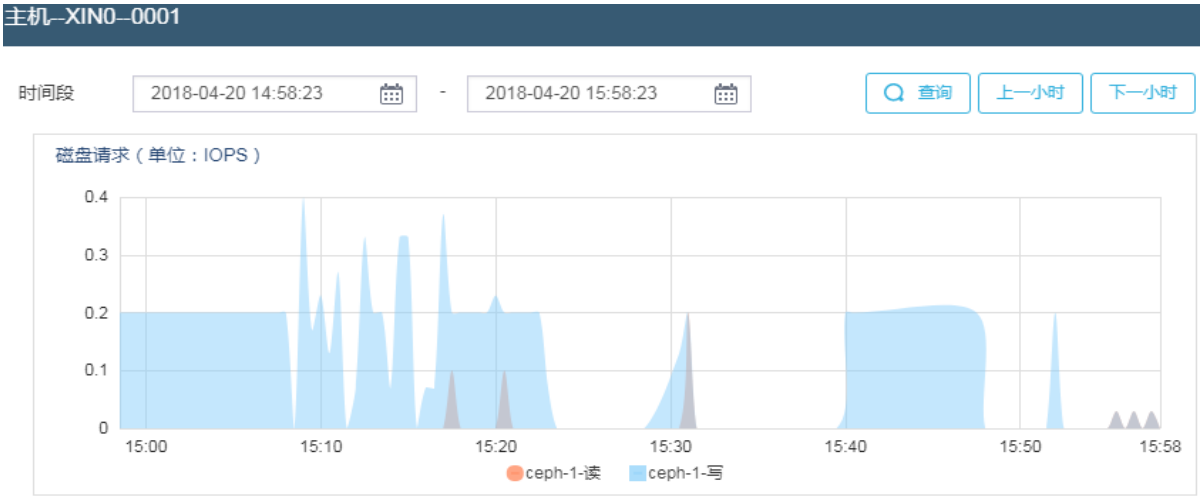
3. 查看主机网络吞吐量

在[性能监控/网络吞吐量统计]页面，点击<更多>按钮可以查看更长时间范围内的各个物理网卡的网络吞吐量信息。



4. 查看主机磁盘利用率

在[性能监控/IOPS]页面显示主机的磁盘占用信息。



5. 查看主机分区占用率

在[性能监控/分区占用率]页面显示主机的分区占用信息。

分区占用率

系统分区	分区类型	分区挂载	容量(GB)	已使用容量(GB)	占用率(%)
/dev/dm-4	ocfs2	/vms/tt	20	4.14	21
/dev/mapper/36...	ext4	/dda	339.46	14.96	5
/dev/mapper/36...	ext4	/ASD	120.95	0.06	1
/dev/mapper/8c...	xfs	/var/lib/ce...	930.54	23.21	3
/dev/mapper/a0...	xfs	/var/lib/ce...	930.54	23.27	3
/dev/sda2	ext4	/	45.71	7.27	17

1.5.4 查看虚拟交换机

检查集群下主机间的虚拟交换机名称是否都是一致的。

在主机的“虚拟交换机”页面，检查各个虚拟交换机状态是否为正常的活动状态，如果状态异常查看物理网卡是否正常。

确认主机的所有虚拟交换机中仅配置一个网关。

1.5.5 查看物理网卡

在主机的“物理网卡”页面检查主机的物理网卡是否正常，包括网卡速率、工作模式、活动状态等。异常情况下会影响到虚拟交换机的性能。

1.6 查看虚拟机

1.6.1 查看 CAS Tools 是否正常运行

查看虚拟机的“概要”页面，确认虚拟机是否安装了 CAS Tools 工具并正常运行。

1.6.2 查看磁盘和网卡类型

1. 查看磁盘类型

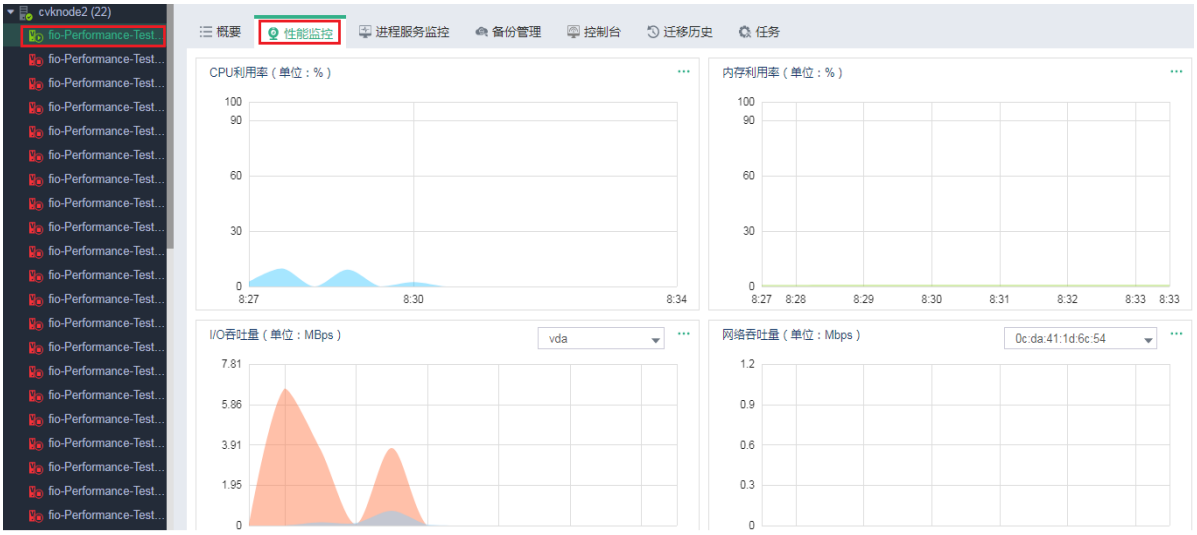
在虚拟机的“修改虚拟机”对话框中的虚拟机磁盘页面，查看设备对象是否为 Virtio 磁盘（提升磁盘性能明显）；源路径是否为共享存储路径；缓存方式是否为“directsync”（建议配置）。

2. 查看网卡类型

在虚拟机的“修改虚拟机”对话框中的虚拟机网卡页面，查看设备型号是否为高速网卡并开启了内核加速（提升网卡性能明显）。

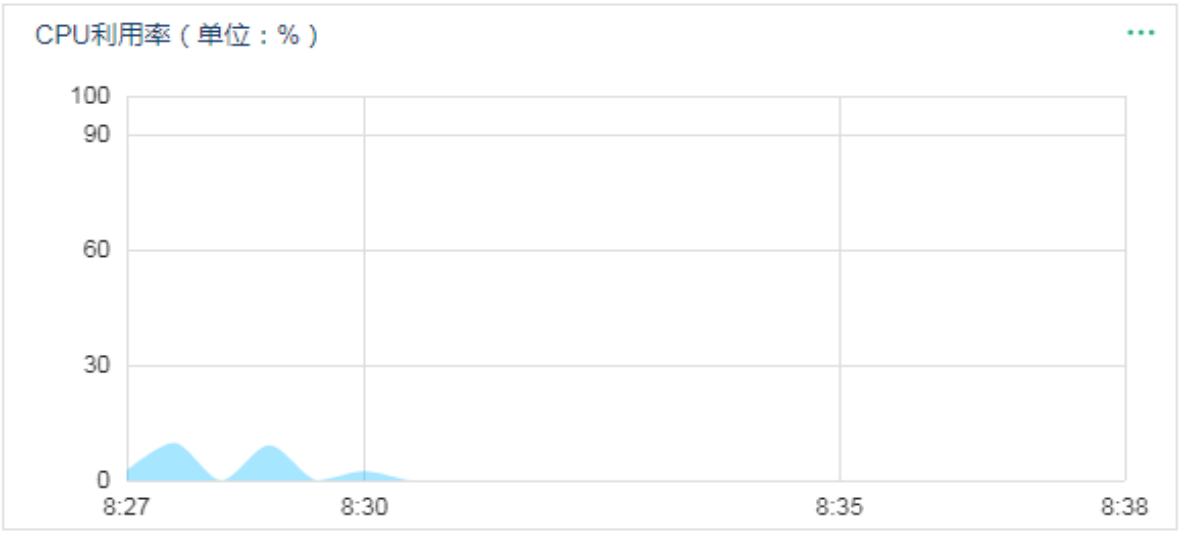
1.6.3 查看虚拟机性能监控

在虚拟机的“性能监控”页面下可以查看到虚拟机的 CPU 利用率、内存利用率、 I/O 吞吐量统计、网络吞吐量统计、磁盘利用率和分区占用率等信息。



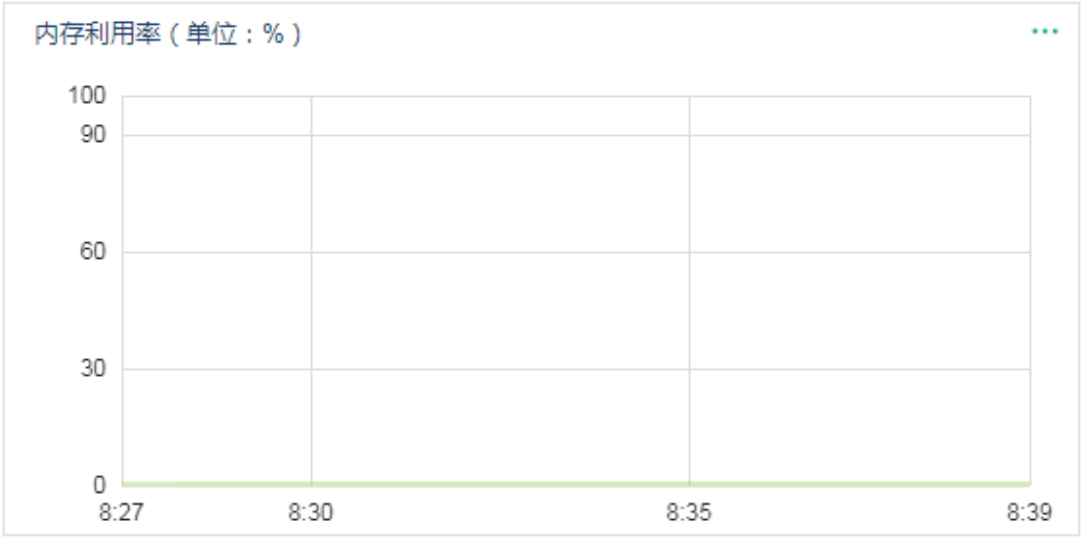
1. 查看虚拟机 CPU 利用率

在[性能监控/CPU 利用率]页面，点击<...>按钮可以查看更长时间范围内的 CPU 占用率信息。



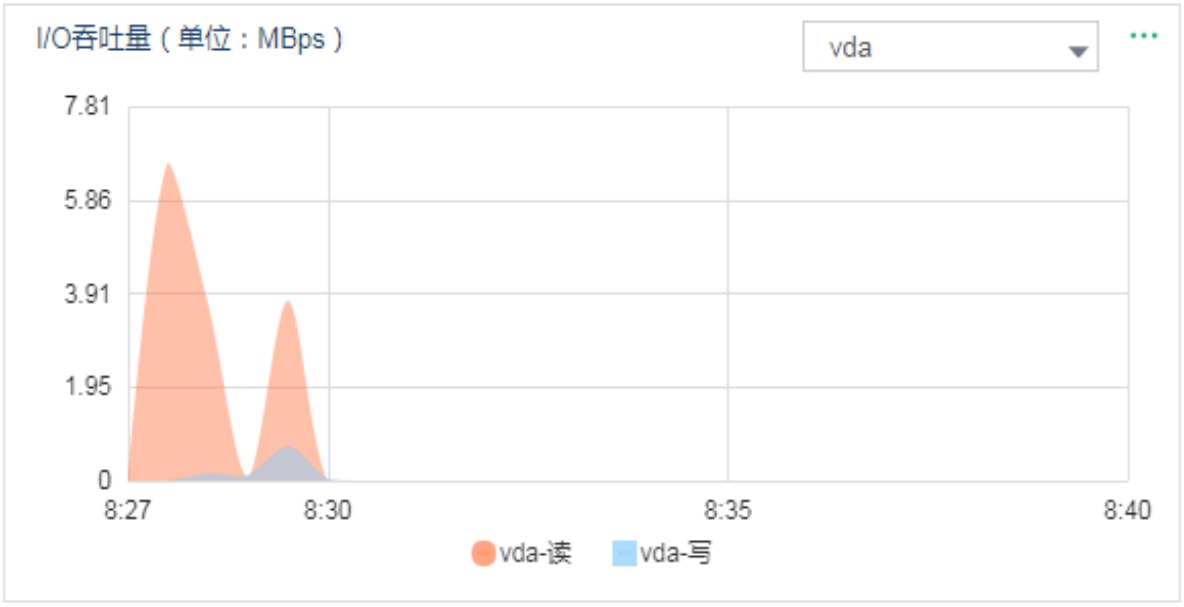
2. 查看虚拟机内存利用率

在[性能监控/内存利用率]页面，点击<...>按钮可以查看更长时间范围内的内存占用率信息。



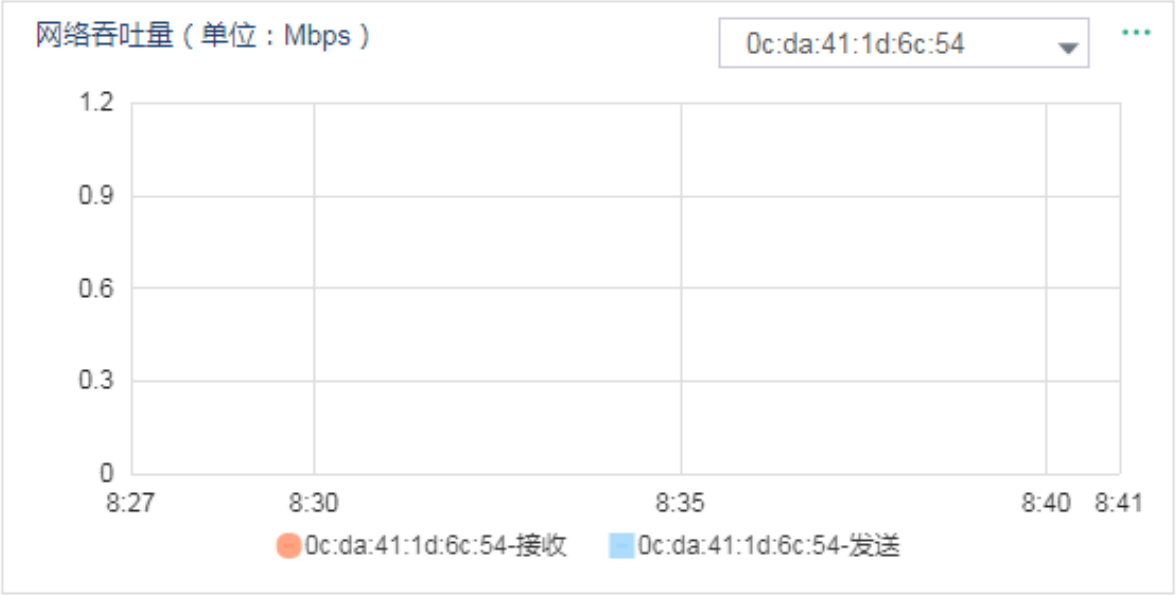
3. 查看虚拟机 I/O 吞吐量

在[性能监控/I/O 吞吐量统计]页面可以查看虚拟机磁盘的 I/O 吞吐量信息，点击<…>按钮可以查看更长时间范围内的 I/O 吞吐量信息。



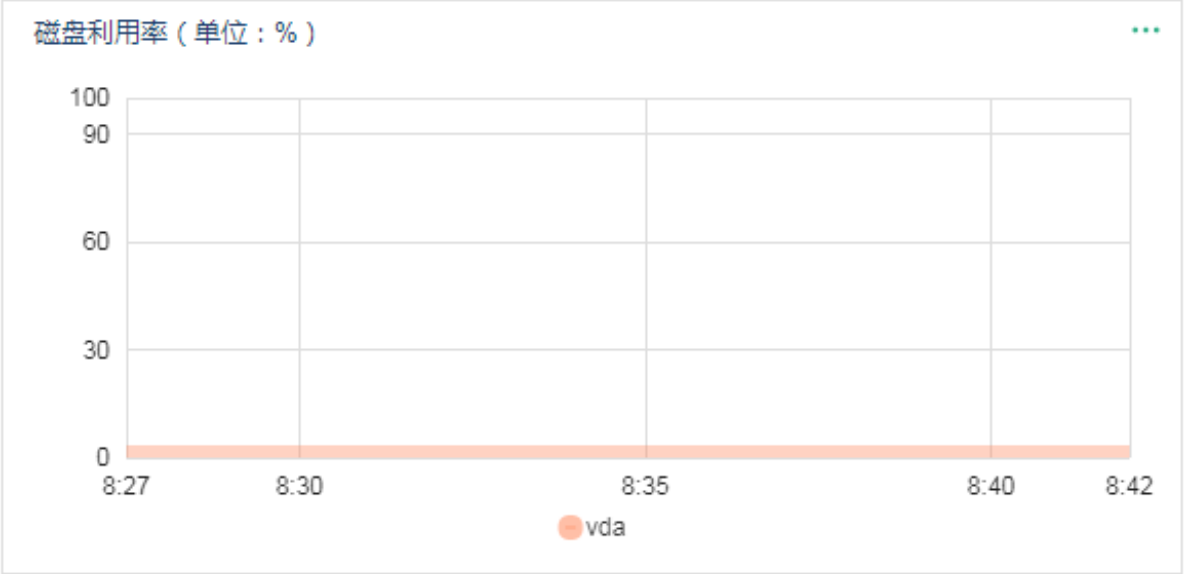
4. 查看虚拟机网络吞吐量

在[性能监控/网络吞吐量统计]页面，点击<…>按钮可以查看更长时间范围内的各个网卡的网络吞吐量信息。



5. 查看虚拟机磁盘利用率

在[性能监控/磁盘利用率]页面显示虚拟机的磁盘利用率信息。



6. 查看虚拟机分区占用率

在[性能监控/分区占用率]页面显示虚拟机的分区占用信息。

分区占用率

系统分区	容量(GB)	已使用容量(GB)	占用率(%)
/dev/mapper/centos-...	141.5	0.04	0.02
/dev/mapper/centos-...	49.98	6.58	13.17
/dev/vda1	0.48	0.15	31.85

1.6.4 查看虚拟机备份

虚拟机备份管理可以查看对应虚拟机的备份历史，UIS 平台上的核心虚拟机建议全部进行备份。

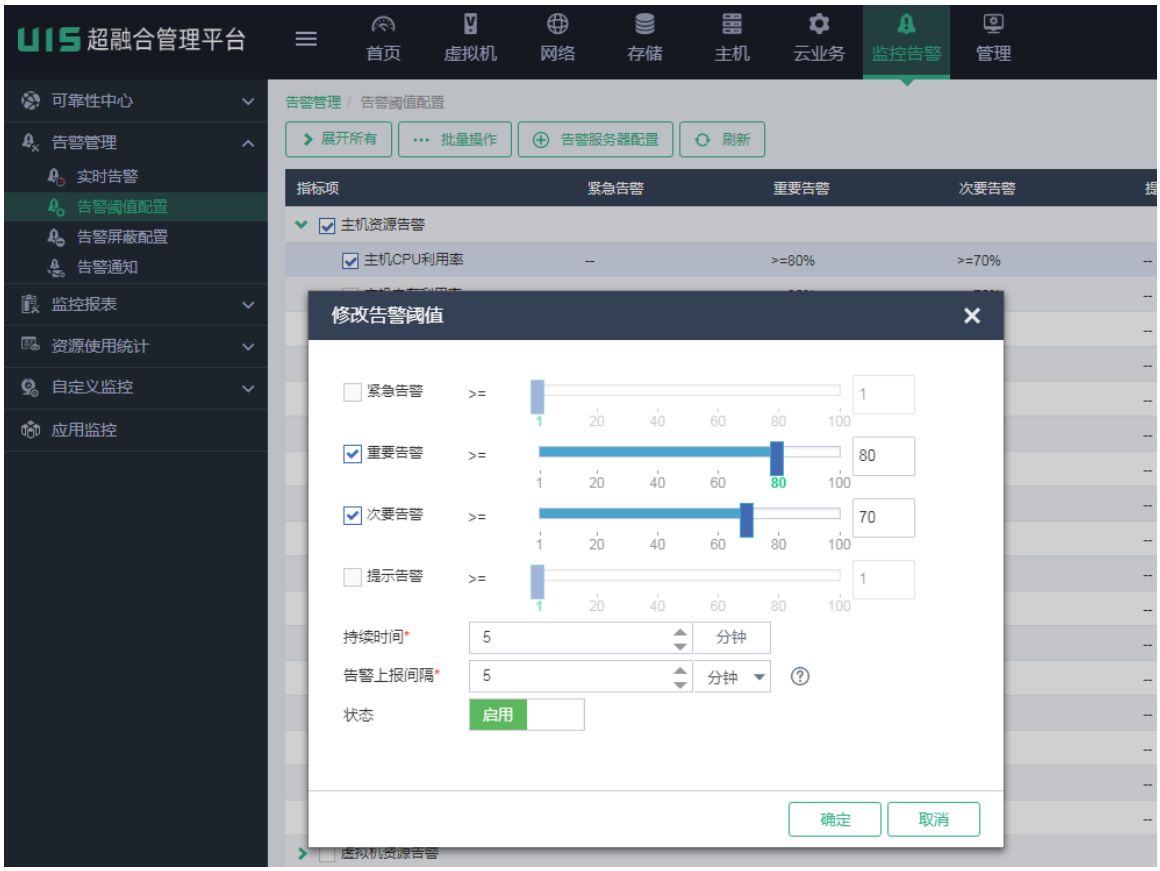
1.7 查看License

UIS 系统主要包含 UIS Manager 组件的 License、CAS 的 license 和分布式存储的 license，正式局点必须使用正式 License，对于测试或者临时局点可以使用临时 License，但是由于临时 License 有使用期限，临时 License 过期后会影响 UIS 系统的正常使用，因此必需提前更新临时 License，防止超期，影响 UIS 系统的正常使用。

如下图是 UIS Manager 组件的 License 界面。

1.8 监控告警管理

告警管理功能用于统计和查看操作员需要关注的告警信息。目前，UIS 统计的告警信息的类型包括主机资源告警、虚拟机资源告警、集群资源告警、故障告警、安全告警、其他异常告警和分布式存储资源告警。



用户可设置对主机或虚拟机的 CPU 利用率、内存利用等指标项的告警阈值。当指标项的实际值达到告警阈值时，将产生告警并上报。用户可以在实时告警列表中查看上报的告警。通过告警屏蔽配置，用户可以将无需关注的告警屏蔽，使其不再上报。系统还支持将告警以邮件或短信的形式发送给用户。

2 UIS 操作风险说明

参见《H3C UIS 超融合管理平台 高危操作手册》

3 日常变更介绍

UIS 系统在运行过程中出现了问题，需要按照一定的规则要求去变更，否则会影响现网业务的正常运行。

3.1 UIS版本升级

请参考版本说明书的“版本升级操作指导”小节完成 UIS 的版本升级操作。

3.2 服务器硬件故障维修

请参考《UIS 超融合一体机部件更换配置指导》

3.3 UIS平台服务器开关机

对 UIS 系统进行整体的维护操作时，需要按照一定的顺序要求进行设备的开机或关闭操作，否则会破坏业务系统。在关机之前需要确认主机的健康度是 100%。

详细参见《H3C UIS 超融合产品标准版正常开关机指导》

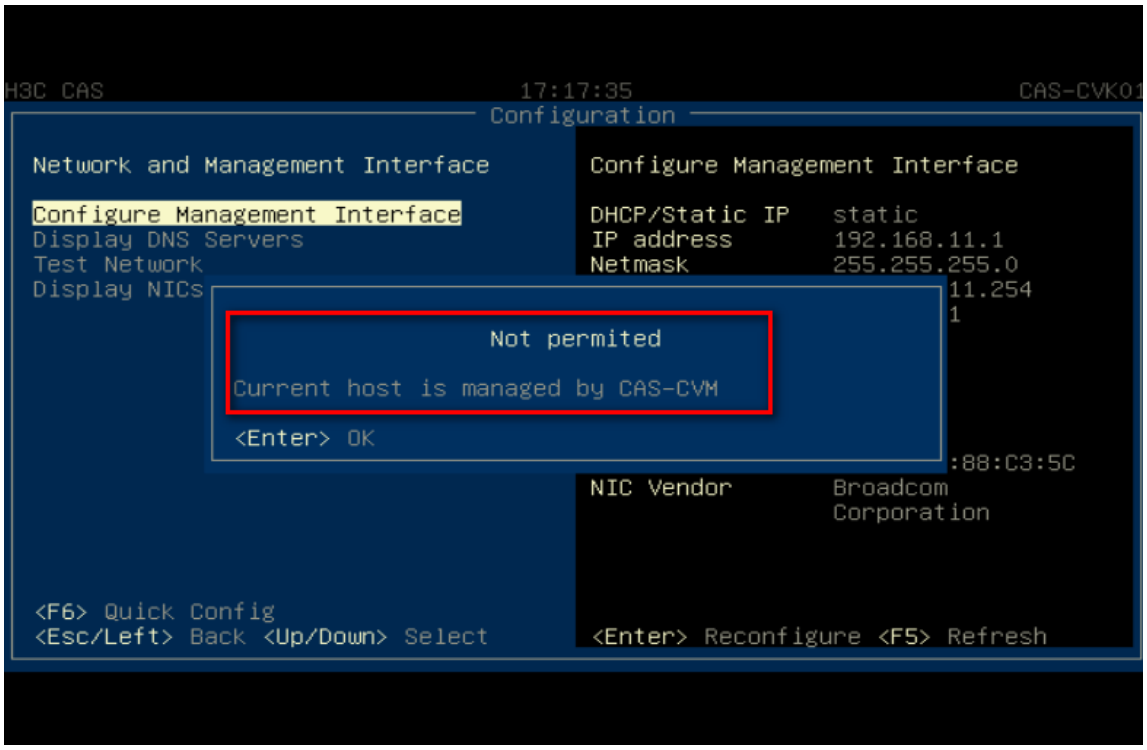
3.4 IP地址和主机名变更

注意

- 不允许在 CVK 后台的命令行中进行 IP 地址和主机的修改操作。
- CVK 的共享存储处于暂停状态时删除 CVK 主机，该共享存储会被自动删除，因此 CVK 主机添加回来后需要重新挂载共享存储。
- 在进行主机删除操作时。系统会自动对数据进行备份操作。
- 管理节点的管理网地址不允许修改。如果错误更改，只能进行重新安装系统。

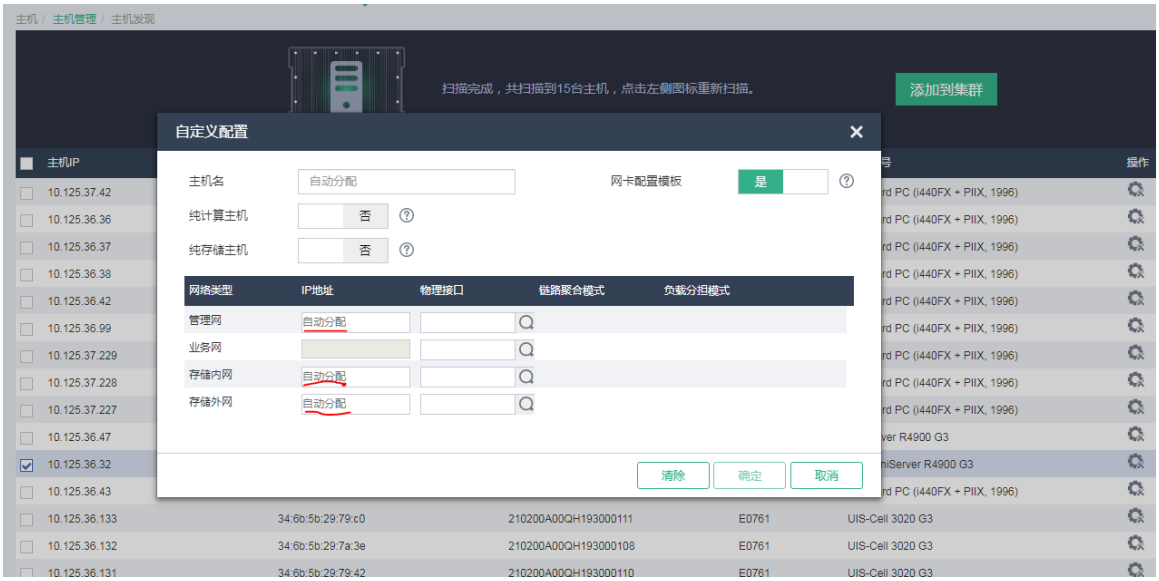
UIS 系统开局完成后，可能会出现变更 UIS 系统 IP 地址或者主机的需求。

CVK 主机添加到 UIS 集群后，无法通过 Xconsole 界面提供的方法修改 IP 地址或者主机名，如下图所示。因此必须先将 CVK 主机从 UIS 系统中删除。



如果 CVK 主机上启用了共享存储或者运行了虚拟机，则无法删除。因此需要先关闭虚拟机（或者迁移虚拟机）、暂停共享文件系统（删除共享文件系统），然后再删除主机。

删除主机后，再次通过扩容主机的形式添加主机，在扩容的过程中，给该主机手动配置相应的 ip 地址并选择相应的网口，然后把主机重新加回集群，再把原来的虚拟机迁移回来。



注意

当节点少于等于 4 台主机时，不支持 IP 地址更改；

填写的 ip 地址要与原有集群内的管理网、存储内外网相通，否则会导致添加主机失败；

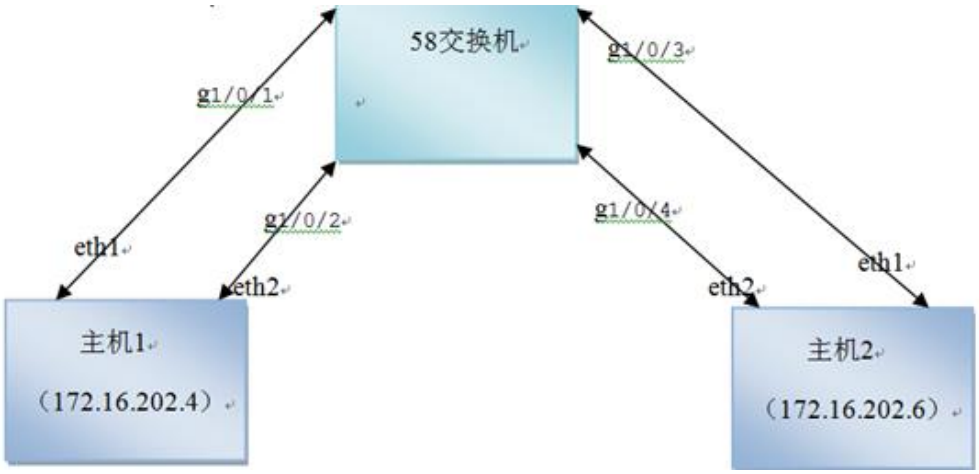
IP 地址的设定属于开局阶段的规划，请在开始阶段做好设定，防止后续不能更换。

3.5 调整管理虚拟交换机绑定的物理接口

现场规划不合理时，需要调整虚拟交换机绑定的物理接口。对已经部署后的环境进行网络变更需谨慎进行，确保熟悉网络拓扑以及变更需求。

3.5.1 调整管理虚拟交换机绑定的物理接口以及端口模式

对虚拟机交换机绑定端口以及模式的修改不能在前台进行，需后台操作。将多个端口聚合在一起形成一个汇聚组，以实现出负荷在各成员端口中的分担，同时也提供了更高的连接可靠性使流量。



链路聚合的优点：

- 增加网络带宽

链路聚合可以将多个链路捆绑成为一个逻辑链路，捆绑后的链路带宽是每个独立链路的带宽总和。

- 提高网络连接的可靠性

链路聚合中的多个链路互为备份，当有一条链路断开，流量会自动在剩下链路间重新分配。

链路聚合根据 bond 口有没有使能 lacp 协议，可分为**静态聚合**和**动态聚合**。

1. OVS 动态聚合

ovs 测和交换机侧均使能 lacp 协议。ovs 的 bond 口上，lacp 有两种配置，active 和 off， active 配置表示使能了 lacp，off 表示去使能 lacp。

动态聚合 lacp_status 状态有三种，分别是 negotiated/configured/disabled 三种状态，negotiated 状态为协商成功状态，configured 状态为 ovs 侧使能 lacp,但是 lacp 协商失败，disabled 状态为 ovs 侧未使能 lacp。

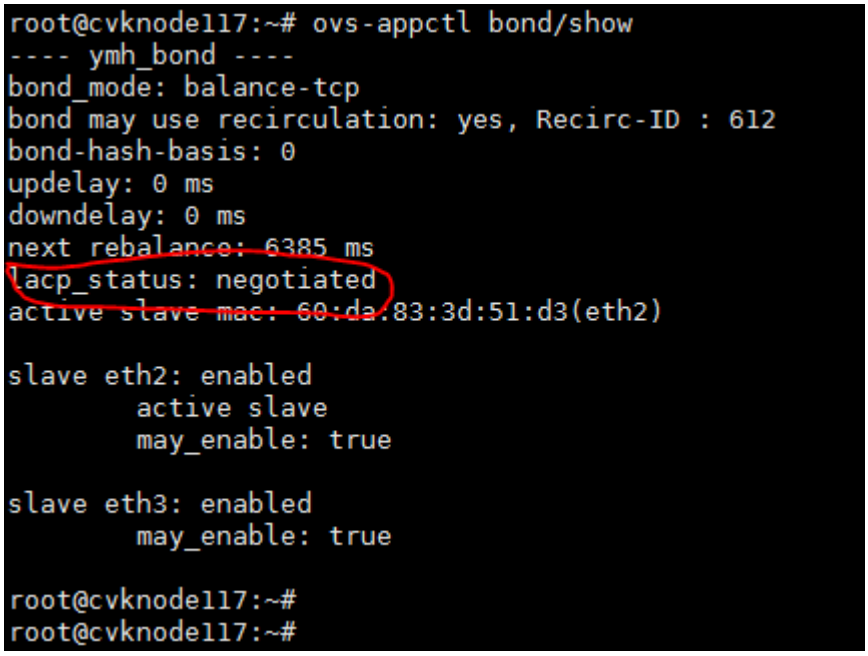
如下面图 1 所示，有 bond 口下面配置 lacp 为 active 状态，ovs 测 bond 口上已经使能了 lacp,但是 bond 的 lacp_status 状态为 configure，这种情况很可能时对端未使能 lacp 导致的。

图1 动态聚合协商失败

```
root@cvknode117:~# ovs-vsctl list Port ymh_bond
_uuid               : 032b03c6-4f59-4570-ac30-7bdc5fe25536
TOS                 : []
bond_active_slave   : "60:da:83:3d:51:d4"
bond_downdelay      : 0
bond_fake_iface     : false
bond_mode           : balance-tcp
bond_updelay        : 0
cvlans              : []
external_ids        : {}
fake_bridge         : false
interfaces          : [31f5424e-c56f-4542-b0b2-1f163cf51cbe, 93ac466b-4868-4c7f-8cf4-cd67323bd40b]
lacp                : active
mac                 : []
name                : ymh_bond
other_config        : {lacp-fallback-ab="true"}
protected           : false
qbg_mode            : []
qos                 : []
rstp_statistics     : {}
rstp_status         : {}
statistics          : {}
status              : {}
tag                 : []
tcp_syn_forbid      : false
trunks              : []
vlan_mode           : []
vm_ip               : []
vm_mac              : []
root@cvknode117:~#
root@cvknode117:~# ovs-appctl bond/show
---- ymh_bond ----
bond_mode: balance-tcp
bond may use recirculation: yes, Recirc-ID : 612
bond-hash-basis: 0
updelay: 0 ms
downdelay: 0 ms
next rebalance: 9790 ms
lacp_status: configured
active slave mac: 60:da:83:3d:51:d4(eth3)

slave eth2: enabled
        may_enable: true

slave eth3: enabled
        active slave
        may_enable: true

root@cvknode117:~#
```

正常情况下，动态聚合 lacp 协商成功，bond 状态如图 2 所示：

图2 动态聚合协商成功



在 ovs 里面，动态聚合又可分为高级负载分担和基本负载分担，这两者的主要区别在于链路 entry 在 hash 过程中，维度不一样。

(1) **balance-tcp 模式**：根据以太网类型，（源,目的）Mac 地址，vlan 号，IP 报文协议，（源,目的）IP 地址（或者 IPv6 地址），（源,目的）四层端口字段进行 hash 得到报文的转发接口；

(2) **balance-slb 模式**：只是根据源 mac 和 vlan 字段进行 hash 得到报文的转发接口，这是当前界面下发的 bond_mode 的配置参数；

2. OVS 静态聚合

ovs 测和交换机侧均**未使能 lacp 协议**，配置成功状态如下：

图3 静态聚合配置状态

```
root@cvknode117:~#
root@cvknode117:~# ovs-vsctl list Port ymh_bond
_uuid               : 032b03c6-4f59-4570-ac30-7bdc5fe25536
TOS                 : []
bond_active_slave   : "60:da:83:3d:51:d3"
bond_downdelay      : 0
bond_fake_iface     : false
bond_mode           : balance-tcp
bond_updelay        : 0
cvlans              : []
external_ids        : {}
fake_bridge         : false
interfaces          : [31f5424e-c56f-4542-b0b2-1f163cf51cbe, 93ac466b-4868-4c7f-8cf4-cd67323bd40b]
lacp                : off
mac                 : []
name                : ymh_bond
other_config        : {lacp-fallback-ab="true"}
protected           : false
qbg_mode            : []
qos                 : []
rstp_statistics     : {}
rstp_status         : {}
statistics          : {}
status              : {}
tag                 : []
tcp_syn_forbid      : false
trunks              : []
vlan_mode           : []
vm_ip               : []
vm_mac              : []
root@cvknode117:~#
root@cvknode117:~# ovs-appctl bond/show
---- ymh_bond ----
bond_mode: balance-tcp
bond may use recirculation: yes, Recirc-ID : 612
bond-hash-basis: 0
updelay: 0 ms
downdelay: 0 ms
next rebalance: 6132 ms
lacp_status: off
active slave mac: 60:da:83:3d:51:d3(eth2)

slave eth2: enabled
        active slave
        may_enable: true

slave eth3: enabled
        may_enable: true

root@cvknode117:~#
```

其中，bond 口配置里面，lacp 状态为 off，聚合 lacp_status 状态为 off。

在 ovs 里面，静态聚合可分为高级负载分担、基本负载分担和主备负载分担。高级负载分担和基本负载分担的区别同动态聚合，下面简单讲述下基本负载分担。

在 ovsdb 中 bond 端口中保存主链路的选择方式，interface 中保存物理网卡的优先级，进行如下配置：

(1) ovs-vsctl set Port bond-name other_config: active-algorithm=”speed|order”

其中，speed 表示按照网卡速率来选择主链路，order 表示按照网卡配置的顺序来选择主链路。此命令不配置时，缺省按照网卡速率来选择主链路。

(2) ovs-vsctl set Port bond-name other_config:active-algorithm=”true|false”

其中，true 表示当选定的主链路网卡 down 了之后又 up 时，会重新切换回去；false 表示不会切换回去。此命令不配置时，缺省不切换回去。

(3) ovs-vsctl set Interface ethx other_config:slave-priority=”n”,

其中 n 为由后台根据用户配置的顺序来分配编号，比如 1,2,3…，数字越小优先级越高。

图4 主备聚合聚合口配置

```
root@cvknode117:~# ovs-vsctl list Port ymh_bond
_uuid               : 032b03c6-4f59-4570-ac30-7bdc5fe25536
TOS                 : []
bond_active_slave   : "60:da:83:3d:51:d4"
bond_downdelay      : 0
bond_fake_iface     : false
bond_mode           : active-backup
bond_updelay        : 0
cvlans              : []
external_ids        : {}
fake_bridge         : false
interfaces          : [31f5424e-c56f-4542-b0b2-1f163cf51cbe, 93ac466b-4868-4c7f-8cf4-cd67323bd40b]
lacp                : off
mac                 : []
name                : ymh_bond
other_config        : {active-algorithm=order, lacp-fallback-ab="true", reselect-on-change="true"}
protected           : false
qbg_mode            : []
qos                 : []
rstp_statistics     : {}
rstp_status         : {}
statistics          : {}
status              : {}
tag                 : []
tcp_syn_forbid      : false
trunks              : []
vlan_mode           : []
vm_ip               : []
vm_mac              : []
root@cvknode117:~#
```

图5 主备聚合 ovs 上成员 interface 上的接口配置

```
root@cvknode117:~# ovs-vsctl list Interface eth2
_uuid               : 31f5424e-c56f-4542-b0b2-1f163cf51cbe
admin_state         : up
bfd                 : {}
bfd_status          : {}
cfm_fault           : []
cfm_fault_status    : []
cfm_flap_count      : []
cfm_health          : []
cfm_mpid            : []
cfm_remote_mpids    : []
cfm_remote_opstate  : []
duplex              : full
error               : []
external_ids        : {}
ifindex             : 4
ingress_policing_burst: 0
ingress_policing_rate: 0
lacp_current        : []
link_resets         : 2
link_speed          : 1000000000
link_state          : up
lldp                : {}
mac                 : []
mac_in_use          : "60:da:83:3d:51:d3"
mtu                 : 1500
mtu_request         : 1500
name                : "eth2"
ofport              : 1
ofport_request      : []
options             : {}
other_config        : {slave-priority="1"}
statistics          : {collisions=0, rx_bytes=3785277, rx_crc_err=0, rx_dropped=6,
ets=1689}
status              : {driver_name=igb, driver_version="5.3.5.4", firmware_version=
type                : ""
root@cvknode117:~#
```

3. OVS 由单网口切换为动态聚合

下面以管理网 vswitch0 由单网口 eth7 切换为 eth5+eth7 的动态聚合高级/基本负载分担举例说明。

(1) 如果 eth5 和 eth7 对端交换机已经配置了动态聚合组，并把这两个口加到了聚合组里面，ovs 侧配置方法：直接在 ovs 侧配置动态聚合高级(bond_mode=balance-tcp)/基本(bond_mode=balance-slb)负载分担即可：

```
ovs-vsctl del-port vswitch0 eth7; ovs-vsctl -- add-bond vswitch0 vswitch0_bond eth5 eth7
bond_mode=[balance-tcp | balance-slb] -- set port vswitch0_bond lacp=active
```

注意

这里";"前后的两条命令必须一起敲，这是为了在管理网断开（vswitch0 踢掉 eth7 口）瞬间马上配置 vswitch0 为 eth5+eth7 的动态聚合模式。

(2) 如果 eth5 和 eth7 对端交换机没有配置动态聚合组，中间使用静态主备聚合来平滑切换聚合模式。

- 在 ovs 侧创建 eth5+eth7 的静态主备聚合模式：

```
ovs-vsctl del-port vswitch0 eth7;ovs-vsctl add-bond vswitch0 vswitch0_bond eth5 eth7
bond_mode=active-backup
```

- eth5 和 eth7 对端交换机配置动态聚合组，并把这两个口加到了聚合组里面（不失一般性，假设 eth5 连接对端交换机口 GigabitEthernet1/0/5，eth7 连接对端交换机口 GigabitEthernet1/0/7）。

```
[H3C]interface Bridge-Aggregation 8                        //创建聚合组 8
[H3C-Bridge-Aggregation8]link-aggregation mode dynamic    //指定聚合组为动态聚合
[H3C]interface GigabitEthernet 1/0/5
[H3C-GigabitEthernet1/0/5]port link-aggregation group 8   //将 G 1/0/5 加入聚合组 8
[H3C]interface GigabitEthernet 1/0/7
[H3C-GigabitEthernet1/0/7]port link-aggregation group 8   //将 G 1/0/7 加入聚合组 8
```

注意

Bridge-Aggregation 8 里面聚合组的配置（尤其是 vlan 的配置）要个聚合组里面各接口（这里是 GigabitEthernet1/0/5 和 GigabitEthernet1/0/7）保持一致，否则动态聚合和静态高级/基本负载分担会出问题。

- 使用如下命令将静态主备聚合配置为动态高级(bond_mode=balance-tcp)/基本(bond_mode=balance-slb)负载分担：

```
ovs-vsctl set port vswitch0_bond bond_mode=[balance-tcp | balance-slb] lacp=active
```

4. OVS 由单网口切换为静态聚合

下面以管理网 vswitch0 由单网口 eth7 切换为 eth5+eth7 的动态聚合高级/基本负载分担举例说明。

(1) 如果 eth5 和 eth7 对端交换机已经配置了动态聚合组，并把这两个口加到了聚合组里面，ovs 侧配置方法：直接在 ovs 侧配置动态聚合高级(bond_mode=balance-tcp)/基本(bond_mode=balance-slb)负载分担即可：

```
ovs-vsctl del-port vswitch0 eth7; ovs-vsctl -- add-bond vswitch0 vswitch0_bond eth5 eth7
bond_mode=[balance-tcp | balance-slb] -- set port vswitch0_bond lacp=active
```

注意

这里";"前后的两条命令必须一起敲，这是为了在管理网断开（vswitch0 踢掉 eth7 口）瞬间马上配置 vswitch0 为 eth5+eth7 的动态聚合模式。

(2) 如果 eth5 和 eth7 对端交换机没有配置动态聚合组：中间使用静态主备聚合来平滑切换聚合模式。

- 在 ovs 侧创建 eth5+eth7 的静态主备聚合模式：

```
ovs-vsctl del-port vswitch0 eth7;ovs-vsctl add-bond vswitch0 vswitch0_bond eth5 eth7
bond_mode=active-backup
```

- eth5 和 eth7 对端交换机配置动态聚合组，并把这两个口加到了聚合组里面（不失一般性，假设 eth5 连接对端交换机口 GigabitEthernet1/0/5，eth7 连接对端交换机口 GigabitEthernet1/0/7）。

```
[H3C]interface Bridge-Aggregation 8                        //创建聚合组 8
[H3C]interface GigabitEthernet 1/0/5
[H3C-GigabitEthernet1/0/5]port link-aggregation group 8   //将 G 1/0/5 加入聚合组 8
[H3C]interface GigabitEthernet 1/0/7
[H3C-GigabitEthernet1/0/7]port link-aggregation group 8   //将 G 1/0/7 加入聚合组 8
```

注意

Bridge-Aggregation 8 里面聚合组的配置（尤其是 vlan 的配置）要个聚合组里面各接口（这里是 GigabitEthernet1/0/5 和 GigabitEthernet1/0/7）保持一致，否则动态聚合和静态高级/基本负载分担会出问题。

- 使用如下命令将静态主备聚合配置为静态高级(bond_mode=balance-tcp)/基本(bond_mode=balance-slb)负载分担：

```
ovs-vsctl set port vswitch0_bond bond_mode=[balance-tcp | balance-slb]
```

5. OVS 由动态聚合切换为静态聚合

下面以 vswitch0 由动态聚合（eth5+eth7）切换为静态聚合举例说明。

动态聚合切换为静态聚合，为了做到尽可能的平滑迁移（尽量减少丢包），中间要借助静态主备聚合。

(1) 将 ovs 侧动态聚合切换为静态主备聚合。

```
ovs-vsctl set port vswitch0_bond bond_mode=active-backup lacp=off
```

(2) 将 eth5 和 eth7 对端交换机口的聚合组去使能 lacp（假设聚合组为 Bridge-Aggregation 8）

```
[H3C]interface Bridge-Aggregation 8
[H3C-Bridge-Aggregation8]undo link-aggregation mode dynamic
```

(3) 将 ovs 侧聚合配置由静态主备切换为静态高级/基本。

```
ovs-vsctl set port vswitch0_bond bond_mode=[balance-tcp | balance-slb]
```

6. OVS 由静态聚合切换为动态聚合

下面以 vswitch0 由静态聚合（eth5+eth7）切换为动态聚合举例说明。

(1) 将 ovs 侧静态聚合切换为静态主备聚合（ovs 测为静态主备的不需要这一步）。

```
ovs-vsctl set port vswitch0_bond bond_mode=active-backup
```

(2) 将 eth5 和 eth7 对端交换机口的聚合组使能 lacp（假设聚合组为 Bridge-Aggregation 8）。

```
[H3C]interface Bridge-Aggregation 8
[H3C-Bridge-Aggregation8]link-aggregation mode dynamic
```

(3) 将 ovs 侧静态主备聚合切换为动态高级/基本负载分担。

```
ovs-vsctl set port vswitch0_bond bond_mode=[balance-tcp | balance-slb] lacp=active
```

7. OVS 上聚合删除

这里以 vswitch0 上 eth5+eth7 动态高级负载分担切换为 eth7 单网口为例举例说明。

(1) 将 vswitch0 侧的聚合模式切换为静态主备聚合：

```
ovs-vsctl set port vswitch0_bond bond_mode=active-backup lacp=off
```

(2) 第二步：将 vswitch0 对端交换机 eth5 和 eth7 口移出聚合组（假设 eth5 连接对端交换机口 GigabitEthernet1/0/5，eth7 连接对端交换机口 GigabitEthernet1/0/7）：

```
[H3C]interface GigabitEthernet 1/0/5
[H3C-GigabitEthernet1/0/5]undo port link-aggregation group
[H3C]interface GigabitEthernet 1/0/7
[H3C-GigabitEthernet1/0/7]undo port link-aggregation group
```

(3) 第将 vswitch0 上静态主备聚合删除，并将 eth7 添加到 vswitch0 里面：

```
ovs-vsctl del-port vswitch0_bond;ovs-vsctl add-port vswitch0 eth7
```

静态高级/基本负载分担切换为单链路与动态高级/基本负载分担切换为单链路方法基本一致，区别是第一步使用的命令为：

```
ovs-vsctl set port vswitch0_bond bond_mode=active-backup
```

注意

聚合之间切换由于各种客观原因限制（比如对端物理交换机限制），CAS ovs 很难做到完全平滑切换，会出现个别丢包的情况，因此模式切换最好选择在用户使用量小的时候进行。

3.6 CVK主机更换磁盘

当集群中某一个磁盘发生故障时，不能直接进行插拔更换，需要结合软件进行相应的操作和配置后才能顺利完成，具体请参考《H3C UIS 硬盘更换指导书》。

3.7 UIS密码修改

注意

不允许在 CVK 后台的命令行中进行 root 用户密码的修改操作。

客户基于用户密码的安全性要求，会定期修改用户密码的需求。下面介绍 UIS root 用户密码的修改方法。

3.7.1 WEB 页面修改主机 root 密码

(1) 右键点击主机，选择【修改主机】按钮。

(2) 在弹出的【修改主机】对话框中输入 root 用户新的密码，并点击【确定】按钮，完成主机密码的修改。

3.7.2 admin 密码修改

uis Manager 页面有一个统一的初始密码。如果需要修改密码的话，可以进入 uis manager 页面之后，在右上角点击 admin 选项，可以更改密码。

3.8 集群扩容

注意

业务上线后，集群扩容操作会触发数据重新平衡，导致集群性能降低，建议在业务量小的情况下扩容，有扩容需求请及时联系总部。同时如果要扩容多台，请逐台扩容，等数据均衡完后再进行下一个节点的扩容。

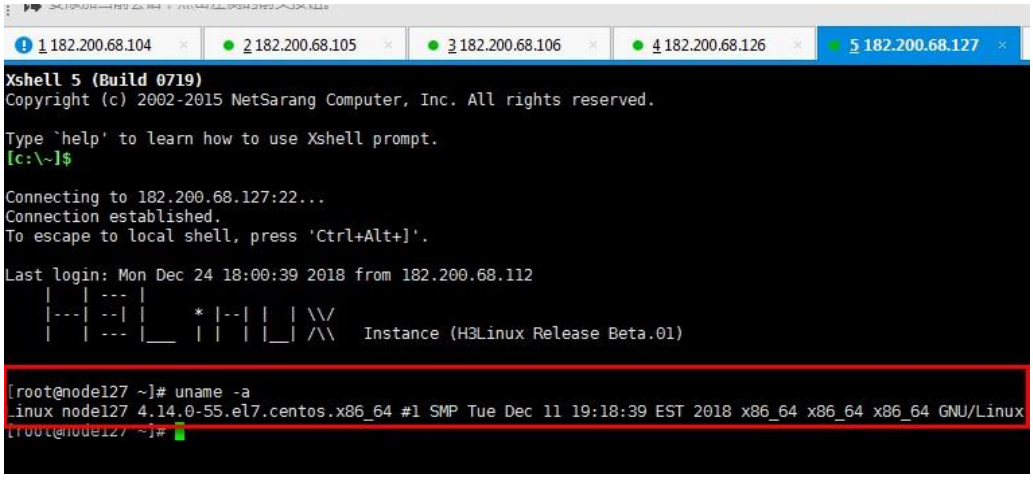
3.8.1 节点扩容

说明

扩容的主机必须与源集群 UIS 版本保持一致,网络配置保持一致。

扩容的主机单盘磁盘容量与原集群节点单盘磁盘容量大小一致，数量差值不大于 1；

(1) 将装好操作系统的服务器管理网、业务网、存储网配置好（与集群节点相应网络在同一网段），保证各个网络层面能通。



(2) 待添加的节点自身必须是没有 ceph 分区且不是其他集群内的节点。

图6 未被使用节点：sd*无 ceph 分区

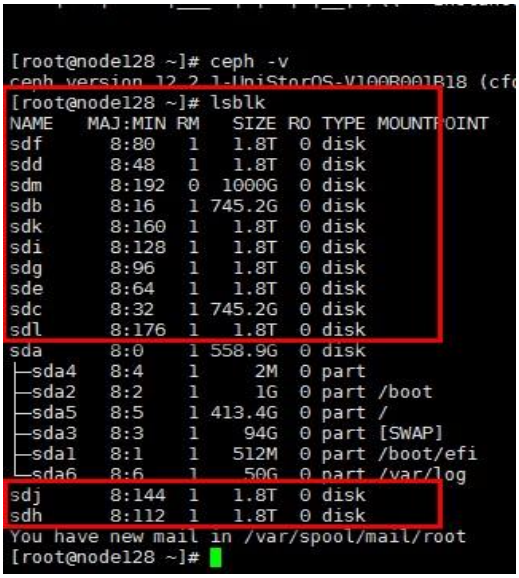
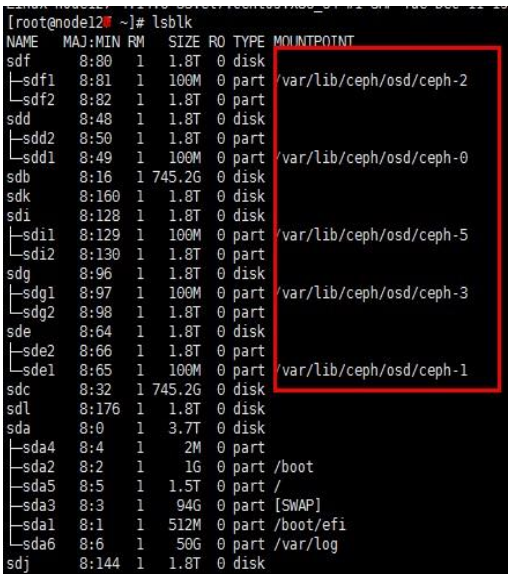


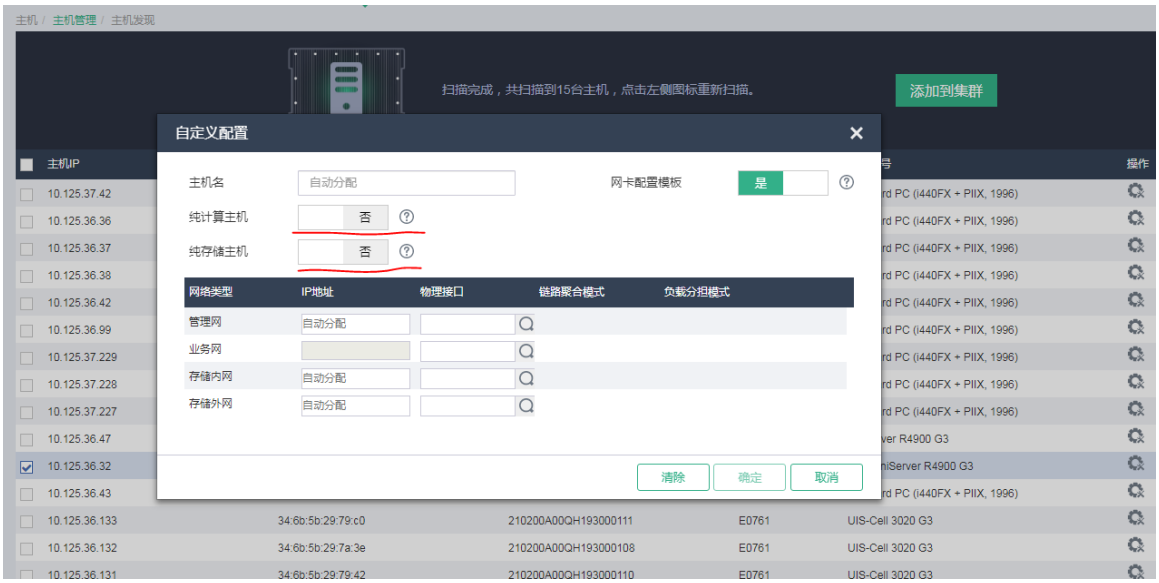
图7 已被使用：sd*有 ceph 分区



(3) 通过主机发现添加主机

(4) 添加扫描主机按钮

(5) 对要添加的主机进行网络配置后，即可将该节点加入到集群中。同时可以选择该主机的角色，支持纯计算节点、纯存储节点，默认不勾选，为融合节点。



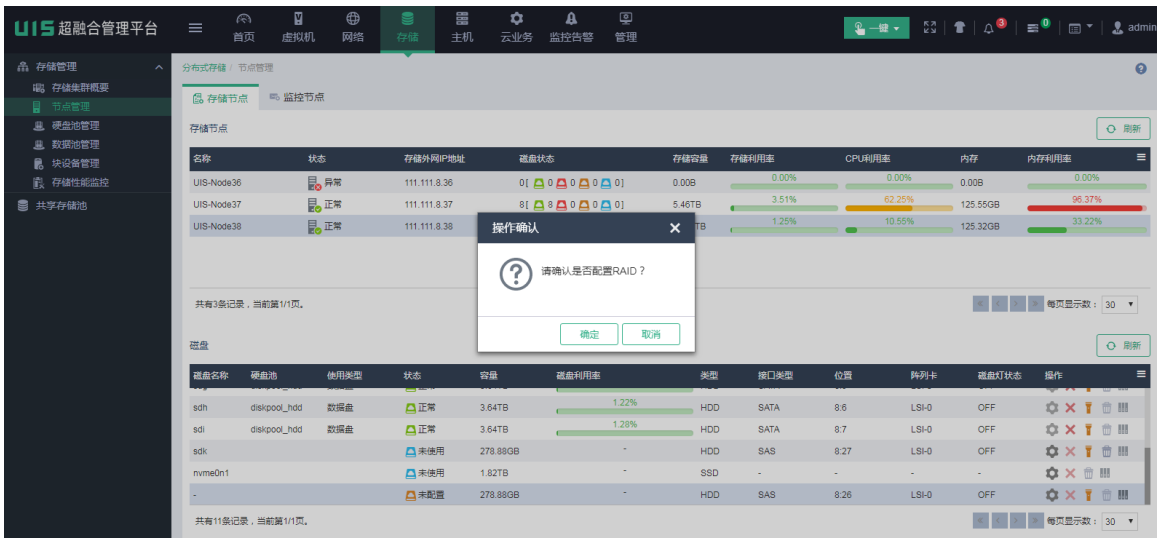
3.8.2 硬盘数量扩容

新增的磁盘必须进行 RAID 后才能加入到集群中而且扩容节点硬盘必须与原集群节点的硬盘容量和数量等规格一致：

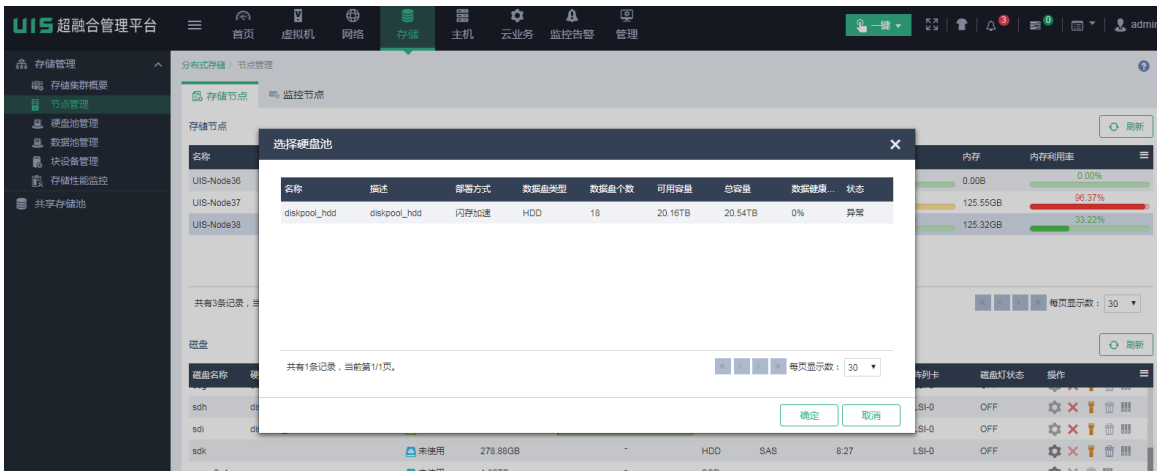
(1) 选择顶部“存储”页签，单击左侧导航树[节点管理]菜单项，进入节点概要页面。

(2) 在存储节点列表中选择目标存储节点。

(3) 在磁盘列表中，选择未配置的磁盘，单击  图标，弹出操作确认对话框，单击<确定>按钮完成操作。



(4) 此时磁盘的状态从“未配置”变成“未使用”状态，单机  图标，选择相应的硬盘池，单机<确认>按钮完成操作。



如果硬盘扩容失败无法自动回退，若硬盘扩容失败需要手动清理残留，针对每一个残留的 osd，在对应的节点后台依次执行以下命令（0 对应 osd id，确保 osd id 正确），清理方法如下：

```
systemctl stop ceph-osd@0.service
umount /var/lib/ceph/osd/ceph-0
rm -rf /var/lib/ceph/osd/ceph-0
ceph osd out 0
ceph osd down 0
ceph osd rm 0
ceph osd crush remove osd.0
ceph osd crush remove device0
ceph auth del osd.0
```

若集群配置了 flashcache 缓存加速需执行以下命令

```
ceph-disk rmfcache --fastremove --fcache28c81f-e89d-487d-9585-6da -- /dev/sd*（假定 fcache28c81f-e89d-487d-9585-6da 为 osd.0 对应的 fcache uuid）
```

若集群采用了元数据分离部署需执行以下命令

```
cat /var/lib/ceph/osd/ceph-0/block.db_uuid(假定输出为 d737d16d-e97e-48a7-8c4c-2f58e904c7f5)
readlink -f /dev/disk/by-partuuid/d737d16d-e97e-48a7-8c4c-2f58e904c7f5（假定输出为/dev/sdf2）
parted -s /dev/sdf rm 2
cat /var/lib/ceph/osd/ceph-0/block.wal_uuid（假定输出为 a87efe76-de8b-4a4b-95a4-d65174c68b3d）
readlink -f /dev/disk/by-partuuid/a87efe76-de8b-4a4b-95a4-d65174c68b3d（假定输出为/dev/sdf5）
parted -s /dev/sdf rm 5
ceph-disk zap /dev/sd*（osd 对应的逻辑盘符）
```

注意

umount /var/lib/ceph/osd/ceph-0 命令会存在失败的情况，是因为 osd 存在服务自动拉起的机制，需要重新执行 systemctl stop ceph-osd@0.service

3.8.3 硬盘容量扩容

本章节**仅适用于不重装系统，对节点数据盘进行更换扩容**（即节点所有数据盘更换为大容量数据盘）。由于操作会涉及到数据迁移，导致集群性能降低，应选择业务量小的时间段操作，避免因集群压力过大，影响业务正常运行，**操作前确保集群健康度 100%**，且无异常告警，三节点由于前台限制，无法删除主机，详情请质询室内。

该方法只支持一次操作一个节点，待数据平衡完毕之后操作下一个。

(1) 删除一个待扩容的存储节点，选择[主机/更多操作/删除主机]。

(2) 确保已经删除对应的数据盘 RAID。

登录对应节点的后台，使用 lsblk 查看，可以看到所有磁盘信息，可以看到数据盘为 sdb，sdc，sdd，sde

```
[root@node124 ~]# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sdd      8:48   0  100G  0 disk
sdb      8:16   0  100G  0 disk
sr0     11:0    1 1024M  0 rom
fd0      2:0    1    4K  0 disk
sde      8:64   0  100G  0 disk
sdc      8:32   0  100G  0 disk
sda      8:0    0  100G  0 disk
├─sda4   8:4    0    1K  0 part
├─sda2   8:2    0    1G  0 part /boot
├─sda7   8:7    0 41.4G  0 part /var/log
├─sda5   8:5    0  512M  0 part /boot/efi
├─sda3   8:3    0 15.8G  0 part [SWAP]
├─sda1   8:1    0    2M  0 part
└─sda6   8:6    0 41.4G  0 part /
You have new mail in /var/spool/mail/root
[root@node124 ~]#
```

确认已经删除数据盘的 RAID。需要注意的是，如果存在数据盘的 RAID 未删除，则需要手动删除 RAID 后再进行后续操作，不要误删系统盘 RAID。

(3) 拔出旧盘，插入新盘

拔出旧盘，插入容量大的新盘，使用 RAID 管理工具对单块磁盘做 RAID 0 操作，注意：关闭物理磁盘的缓存，开启 RAID 卡的缓存。具体关闭开启方法参考开局指导书。

使用 lsblk 命令检验，查看是否所有数据盘均能被识别。

(4) 添加该节点到集群中

通过主机发现方式，添加主机，步骤详情节点扩容章节。

(5) 等待数据平衡

查看告警信息或者输入 ceph –s，当集群健康度为 100%之后，对另一个节点重复 1-5 步 ，直到所有节点的数据盘均更换为大容量硬盘。

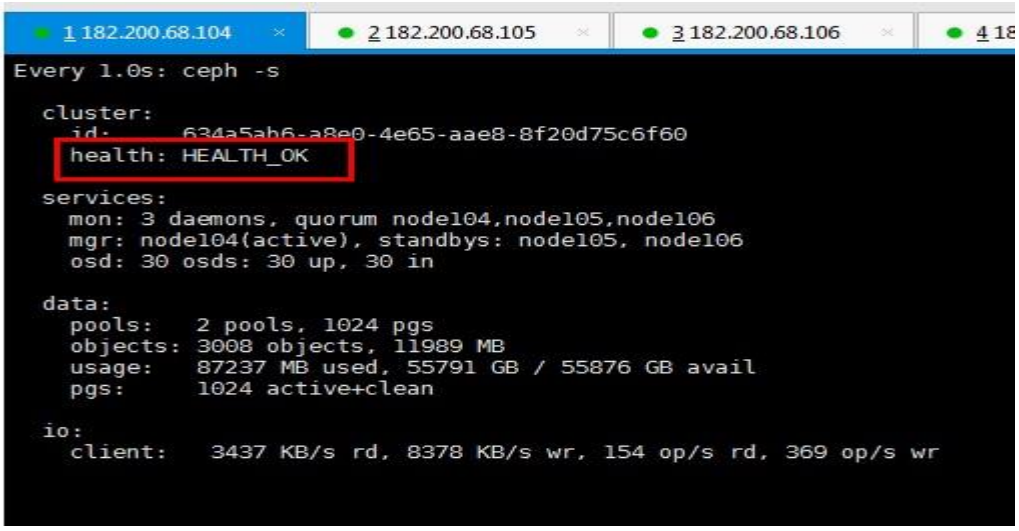
说明

数据量大，数据平衡会消耗大量时间，建议停业务操作。

3.9 集群缩容

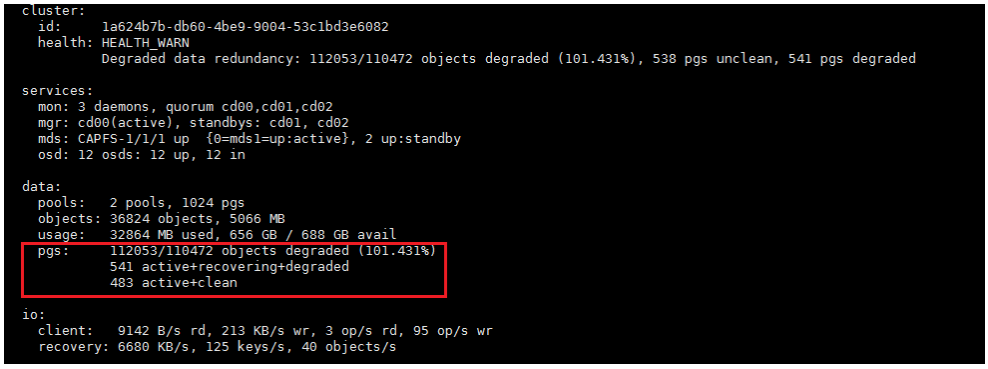
3.9.1 注意事项

(1) 业务上线后，集群缩容操作会触发数据重新平衡，导致集群性能降低，建议在离线的情况下执行缩容，如果有在线缩容的需求，请及时联系总部！！！缩容前确保集群健康度 100%，无异常告警。



(2) 集群内维护模式下的节点无法进行缩容操作。

(3) 集群缩容时，需要保证 PG 状态正常，否则无法操作。



(4) 查看待删除磁盘/主机所在的硬盘池的容量，如果使用率达到了 85%。则不能执行删除操作。

(5) 当前集群节点数为 4，不允许缩容。

3.9.2 缩容方式简介

- 保证管理网、业务网、存储网配置各个网络层面能通。
- 在 UIS 界面进行缩容操作，方式有两种即节点缩容和硬盘缩容。

3.9.3 节点缩容

选择选择主机-更多操作-删除主机，删除主机后需要等待一段时间，集群健康度 100%完成后才能删除其它的主机，删除后的集群应该满足最小 3 节点的要求。

注意

删除主机节点需要保证：删除后，该主机所在【节点池的副本数】<=【节点池主机个数】，否则无法删除。

3.9.4 硬盘缩容

(1) 选择顶部“存储”页签，单击左侧导航树[节点管理]菜单项，进入节点概要页面。

(2) 在存储节点列表中选择目标存储节点。

(3) 在磁盘列表中，选择未配置的磁盘，单击 图标，弹出操作确认对话框，单击<确定>按钮完成操作。

删除硬盘后需要等待一段时间，集群健康度 100%后才能删除其它的硬盘。

需要注意的是，如果主机只有一块硬盘，则无法通过硬盘缩容的方法删除该硬盘。需要采取删除主机的方式，进行缩容。

3.10 NTP时间修改

3.10.1 注意事项：

- 虚拟机如果开启时间同步功能会自动将虚拟机内部时间修改为虚拟机所在主机的系统时间，会对虚拟机内部业务造成影响，故修改时间前必须关闭所有虚拟机时间同步功能;
- 严禁向过去修改时间。该行为会导致 web 页面被禁止登陆、文件系统异常服务器无法正常启动、多个功能逻辑混乱、显示错误等异常，可能导致 H3C UIS 管理平台出现未知异常;
- 向未来修改时间，License 的有效期会相应缩短，注意提前计算 License 失效时间或准备新的 License，避免 License 失效无法登陆。
- 向未来修改时间，跳过时间段对应的主机/虚拟机的性能监控数据、报表数据会缺失。修改时间后 web 页面主机/虚拟机性能监控页面可能显示“暂无数据”，需等待一段时间获取相关数据。

- 向未来修改时间，访问策略的访问时间控制、双因子（CRL 定时更新）、密码策略有效期、开关机策略生效时间、ACL 策略启用时间段、备份策略、快照策略等定时功能将受到影响，功能生效时间将以修改后的为准，请提前评估是否需要修改相关配置或停止相关功能。
- CVM 双机环境，修改时间过程中注意保证主备 CVM 时间一致，NTP 同步时间慢时建议手动修改，具体修改方法见步骤 7。

3.10.2 服务器向未来修改时间操作步骤

请仔细阅读注意事项，了解修改系统时间带来的影响和风险。如果可以接受，再继续下一步骤。

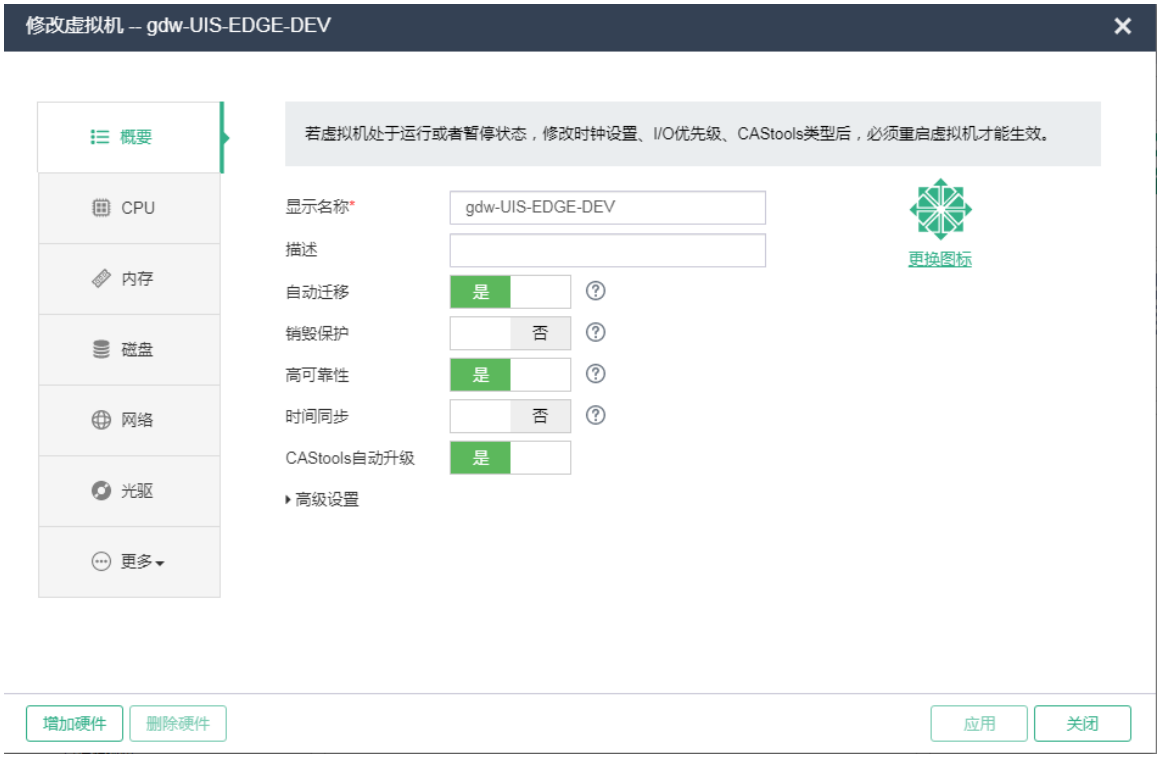
(1) 备份 CVM 配置。

(2) 检查是否启用以下功能：访问策略生效时间段、双因子的 CRL 定时更新、密码策略有效期、ACL 策略生效时间段、CVM 配置定时备份、开关机策略、备份策略、快照策略。如果有配置请检查是否需要根据修改时间情况重新调整功能生效时间、停止相关功能。

(3) 同时使用’ntpq -p’查看各个主机的 NTP Server 是否与配置的一致，而且 offset 后数字的绝对值小于 0.15 sec 。


```
[sysadmin@cvknode2 ~]$ ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*cvknode1        LOCAL(0)         5 u    8   16  377    1.820    0.041   1.100
[sysadmin@cvknode2 ~]$ exit
登出
Connection to cvknode2 closed.
[sysadmin@cvknode1 ~]$ ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*LOCAL(0)        .LOCL.           4 l   12   16  377    0.000    0.000   0.000
```

关闭所有虚拟机时间同步功能。



(4) 暂停集群中的共享存储

(5) 正常关闭集群中的虚拟机。

(6) Web 页面检查确保虚拟化平台无正在运行中的任务，确认后在后台修改时间。date 命令修改系统时间，如 date -s '2019-1-1 12:00:00'；hwclock 命令修改服务器硬件时间，如 hwclock –w 将系统时间同步到硬件时间、hwclock --set --date='2019-1-1 12:00:00'。修改完成后检查各服务器系统时间、硬件时间。date 命令检查系统时间是否修改无误，hwclock 命令检查服务器硬件时间是否修改无误，offset 后数字的绝对值小于 0.15 sec。

(7) 启动共享存储以及虚拟机，恢复业务。

注：如果集群配置为外部时钟，NTP Server 也需要同步修改为和 UIS 服务器相同的时间。

3.10.3 时区修改操作步骤

时区修改可以使用命令 timedatectl set-timezone [ZONE]，例如印度尼西亚

timedatectl set-timezone Asia/Jakarta

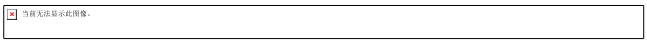
```
[root@cvknode1 uis_mk_cluster_timezone]# timedatectl set-timezone
Display all 426 possibilities? (y or n)
Africa/Abidjan                 America/Atikokan              America/Menominee             Asia/Amman
Africa/Accra                   America/Bahia                 America/Merida                Asia/Anadyr
Africa/Addis_Ababa             America/Bahia_Banderas        America/Metlakatla            Asia/Aqtau
Africa/Algiers                 America/Barbados              America/Mexico_City           Asia/Aqtobe
Africa/Asmara                  America/Belem                 America/Miquelon              Asia/Ashgabat
```

可以选择在每台界面执行修改，也可以通过脚本执行，例如以修改为印度尼西亚时间为例。

uis_mk_cluster_timezone.tar.gz

1. 下载并上传到 cvm 节点后台

2. 解压附件文件



3. 进入解压后的目录，并给脚本添加可执行权限

```
cd uis_mk_cluster_timezone/
_mk_cluster_timezone]# chmod +x uis_mk_cluster_timezone.sh
```

注：如果要修改为其它时区，只需要更改脚本中的时区名称即可

```
#!/bin/bash
str="set-timezone Asia/Jakarta"
cmd=${cmd:-"date;echo $str;timedatectl set-timezone Asia/Jakarta;date"}
```

4. 执行脚本

```
./uis_mk_cluster_timezone.sh
```

脚本会罗列出集群的所有节点，如果包括了所有节点，则输入 yes 继续进行，否则输入 no 退出。

```
node=10.125.36.166
node=10.125.36.132
node=10.125.36.133
node=10.125.36.136
node=10.125.36.135
node=10.125.36.131
node=10.125.36.130
Check node list ok, set timezone now?[yes/no]:
```

输入 yes 后，会对集群各个节点进行时区更改，并输出前后设置的 date 显示。

```
yes
---node=10.125.36.166
Warning: Permanently added '10.125.36.166' (ECDSA) to the list of known hosts.
Authorized users only. All activity may be monitored and reported
2021年 08月 21日 星期六 10:40:52 CST
set-timezone Asia/Jakarta
2021年 08月 21日 星期六 09:40:52 WIB
---node=10.125.36.132
Warning: Permanently added '10.125.36.132' (ECDSA) to the list of known hosts.
Authorized users only. All activity may be monitored and reported
2021年 08月 21日 星期六 10:30:01 CST
set-timezone Asia/Jakarta
2021年 08月 21日 星期六 09:30:01 WIB
```

5. 手动检测各个主机时区是否设置正确。

6. 各个主机后台时区设置完成后，重启 tomcat8 服务(双机场景，只在当前主节点执行)

```
systemctl restart tomcat8.service
```

7.重新登录前台查看主机的时间

3.11 异构/同构迁移

参见《UIS 超融合云迁移方案最佳实践》。

3.12 虚拟机的重定义变更

在某些情况下，例如虚拟机所在的主机异常，上面的虚拟机无法启动，需要在其他主机上重新 define 虚拟机并拉起

3.12.1 查询虚拟机的 xml

在开启了 HA 的情况下，虚拟机的 xml 会默认在 CVM 主机的 HA 目录下保存一份，一般来说位置在/etc/cvm/ha/*clust_id/cvk_name* 下，例如：/etc/cvm/ha/2/cvknode191。在对应的目录下口找到虚拟机所在 cvk 的目录，进入该目录会有对应的虚拟机，例如 test01 虚拟机的 xml。

```
root@cvknode-98:/etc/cvm/ha/1/cvknode-98# ll
total 40
drwxr-xr-x 4 root root 4096 Jun 19 09:59 ./
drwxr-xr-x 3 root root 4096 Jun 19 09:59 ../
-rw------- 1 root root 6554 Jun 17 10:56 CVM01.xml
-rw------- 1 root root 6541 Jun 17 11:03 CVM02.xml
drwxr-xr-x 2 root root 4096 Jun 19 09:59 domainProfile/
drwxr-xr-x 2 root root 4096 Jun  1 14:41 snapshot/
-rw-r--r-- 1 root root 6945 Jun 19 09:59 test01.xml
```

3.12.2 查询虚拟机的磁盘文件所在的存储卷

如果现场本身就是知道虚拟机磁盘所在存储卷的，那就在其他挂载了该存储卷的的主机的后台，确认对应的存储卷是否正常；如果不清楚所在存储卷的，可以通过 1 里面虚拟机的 xml 来确认，通过 vim 或者 cat 查看该 xml，找到对应磁盘的位置，例如

```
<disk type='file' device='disk'>
  <driver name='qemu' type='qcow2' cache='directsync' io='native'/>
  <source file='/vms/images/test01'/>
  <target dev='vda' bus='virtio'/>
  <hotpluggable state='on'/>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x0a' function='0x0'/>
</disk>
```

Source file 所显示的路径就是磁盘文件所在的具体位置。

3.12.3 拷贝虚拟机 xml 到对应的主机下

将 3.12.1 中的 xml 通过 scp 的方式拷贝到在 2 中确认了存储卷位置的主机的/etc/libvirt/qemu 目录下。

3.12.4 通过 xml 进行虚拟机 define 操作

在/etc/libvirt/qemu 目录下执行 virsh define *vm_xml* 的操作，如图

```
root@cvknode-98:/etc/libvirt/qemu# virsh define test01.xml
Domain test01 defined from test01.xml
```

可以看到虚拟机通过 xml 被 define 起来

后台 virsh list –all 也能看到该虚拟机

```
root@cvknode-98:/etc/libvirt/qemu# virsh list --all
 Id    Name      State
------------------------
 1     CVM01     running
 2     CVM02     running
 -     test01    shut off
```

在 CVM 前台对该主机进行连接主机操作，就可以在前台看到该虚拟机，并可以通过前台启动。

在需要 define 的虚拟机数量比较多的情况下，还可以通过重启 libvirt（确认没有中文虚拟机）的方式自动 define 虚拟机，如图所示，在 define 成功后，在前台启动虚拟机。

```
root@cvknode-98:/etc/libvirt/qemu# service libvirt-bin restart
 * Restarting libvirt management daemon /usr/sbin/libvirtd
 * old pid: 4317
 * new pid: 23642
root@cvknode-98:/etc/libvirt/qemu# virsh list --all
 Id    Name      State
-------------------------
 1     CVM01     running
 2     CVM02     running
 -     test01    shut off
```

3.12.5 清理原主机上的虚拟机

要分不同的情况：

- 如果确认主机已经因为某些硬件原因完全损坏，那么建议是将该服务器硬件修复后重新安装和原有系统相同的 UIS 版本。
- 如果主机不是硬件有问题，需要在故障主机启动前先拔掉网线，登录主机后台删除虚拟机 xml 文件，防止重启服务器后 HA 将源主机上的虚拟机拉起，造成双写；

3.13 单机改双机

请参见《H3C UIS 超融合产品双机热备配置指导》。

注：如果 ONEStor 界面无法打开，请登录 cvm 节点，执行如下命令

```
mv /opt/h3c/webapp/content/dsm/index.html.bak /opt/h3c/webapp/content/dsm/index.html
```

执行完变更后，请执行如下命令：

```
mv /opt/h3c/webapp/content/dsm/index.html /opt/h3c/webapp/content/dsm/index.html.bak
```

3.14 SSD缓存容量修改

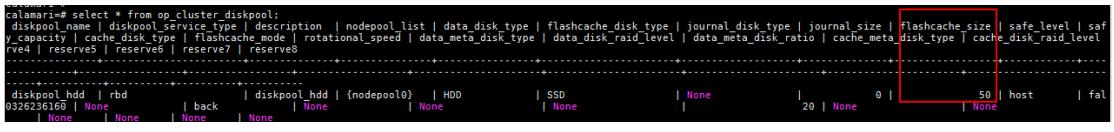
3.14.1 修改缓存分区大小

举例修改 ONEStor 数据库缓存分区大小为 200G，方法如下：

在管理节点执行 onestor cm query -t handyha 确认 ONEStor 双机的主节点；

sudo -u postgres psql calamari; 进入数据库

select * from op_cluster_diskpool; 查询当前 pool 对应的缓存分区大小



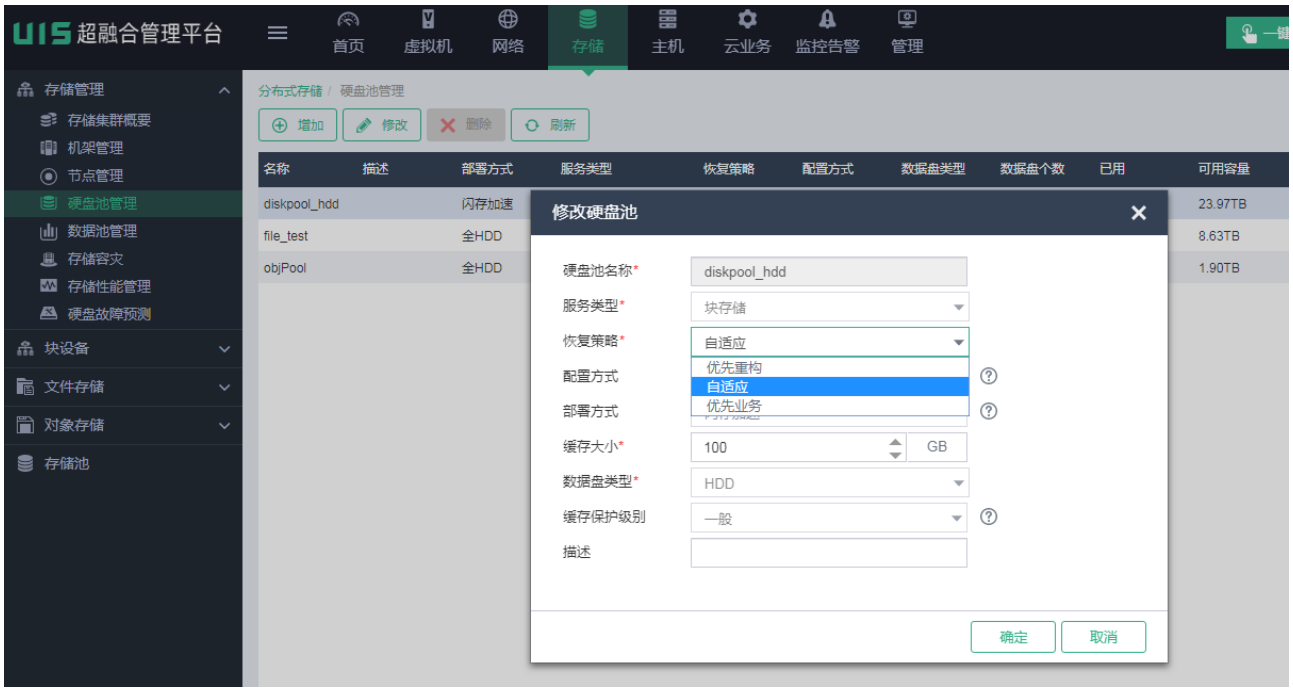
update op_cluster_diskpool set flashcache_size=200 where diskpool_name='diskpool_hdd';

(硬盘池名称待确认)

3.14.2 修改数据平衡优先级

- UIS前台调整，操作如下

首先选中需要当前发生了数据均衡的硬盘池，再点击“修改”,出现如下对话框



“恢复策略”由“自适应”改成“优先重构”。

3.14.3 更改副本数与最小副本数

- 后台修改数据池的副本数，由 3 副本修改为 2 副本，最小副本数改为 1

  ceph osd pool ls detail 查看数据池信息

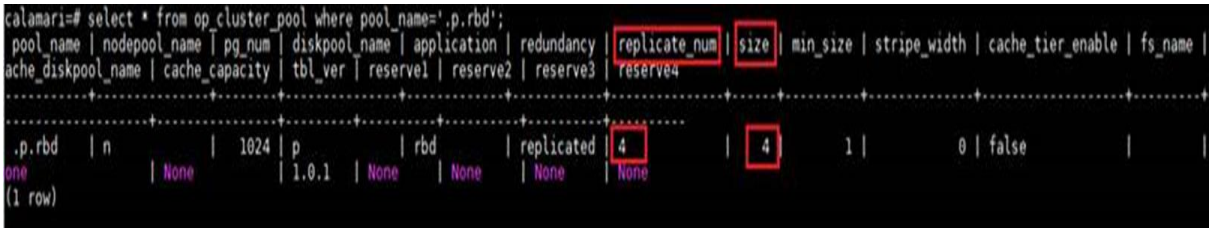
  ceph osd pool set xxx size 2（xxx 为池的名字）

  ceph osd pool set xxx min_size 1（xxx 为池的名字）

- 在 handy 节点（主节点）修改数据库，修改池的副本数，由 3 副本改为 2 副本

✓ 进入 postgres 数据库

✓ 使用 select * from op_cluster_pool where pool_name=’元数据池名称’

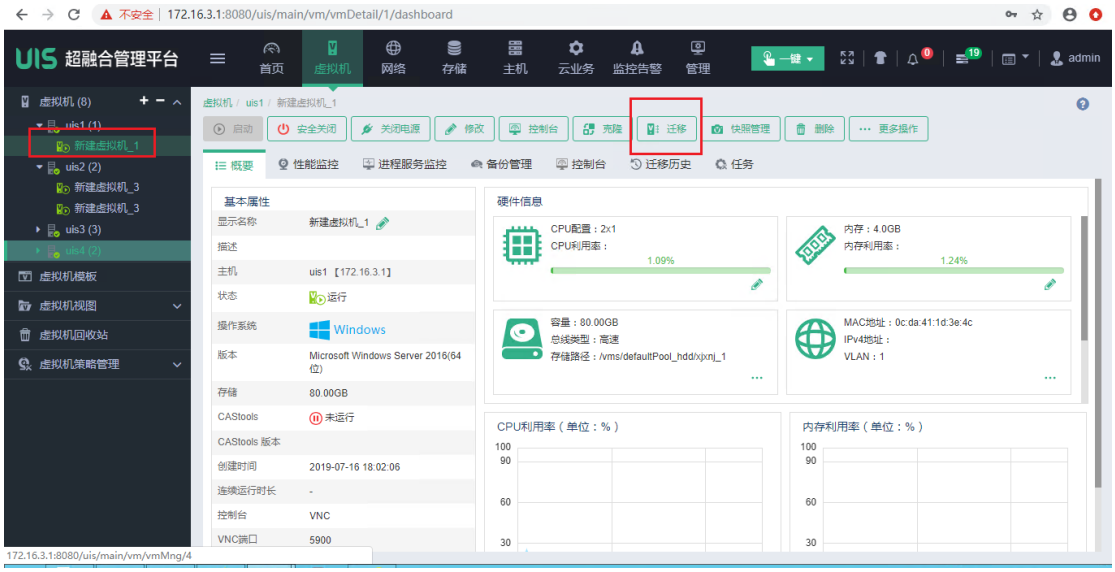


此时数据池的 replicate_num 和 size 值应为 2.

✓ 使用 update op_cluster_pool set size=2,replicate_num=2 where pool_name='池名称';修改副本数

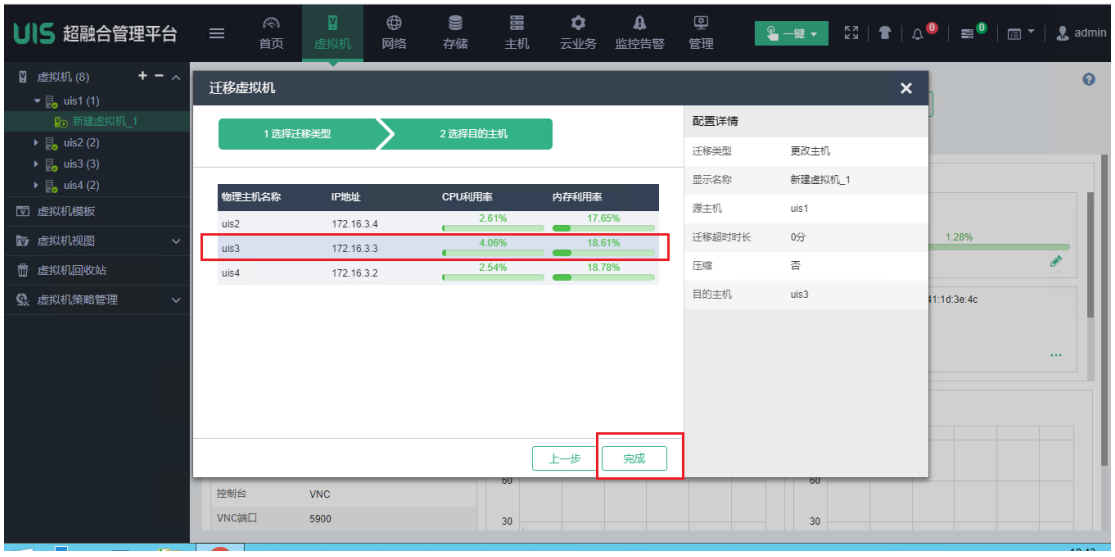
3.14.4 进行虚拟机迁移

选择需要删除的主机下的虚拟机，点击迁移（注意：我们最后迁移管理节点主机的虚拟机）：



选择迁移类型为更改主机，迁移超时时长为 0，点击下一步：

选择需要迁移的主机，点击完成：



迁走虚拟机后，需要将待删除主机的共享存储暂停并删除，保证该主机上没有 iscsi 会话。执行 tgt-admin -s | grep Initiator，检查是否还有会话。

3.14.5 修改 osd_max_backfills 参数

- 点击删除节点前，执行以下命令修改 osd_max_backfills 参数。如果集群容量使用不超过 50%，可以不执行该步骤。

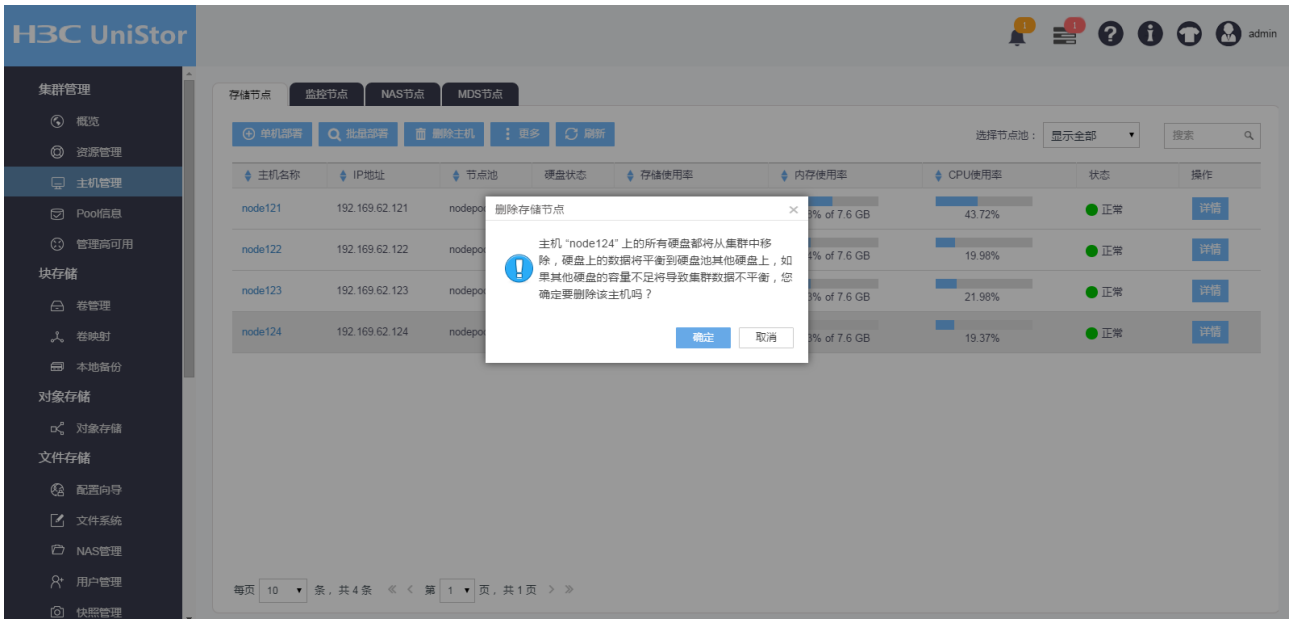
Ceph tell osd.* injectargs --osd_max_backfills=2

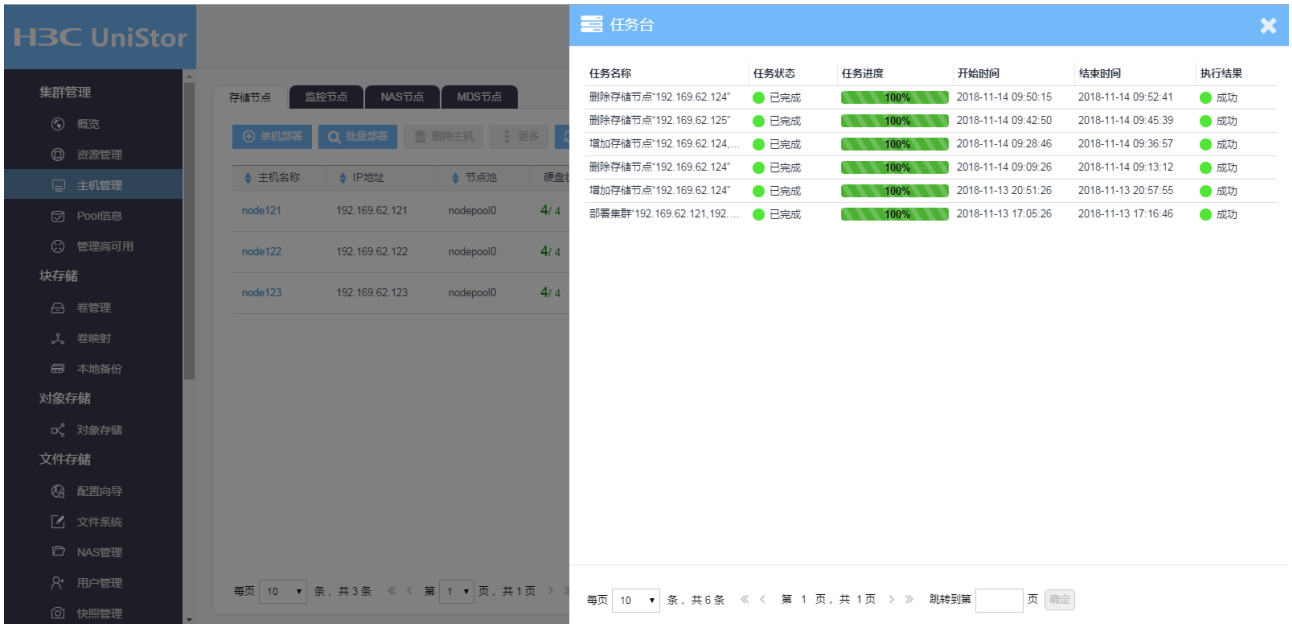
```
[root@node6 ~]# ceph tell osd.* injectargs --osd_max_backfills=2
osd.0: osd_max_backfills = '2'
osd.1: osd_max_backfills = '2'
osd.2: osd_max_backfills = '2'
```

- 每次删除节点前都要再执行一下上面的命令，因为新加的节点还是默认配置。

3.14.6 在 ONEStor 界面删除对应主机的存储角色

选择 集群管理==》主机管理==》存储节点==》选择节点==》删除主机，删除主机后需要等待一段时间，集群健康度 100%完成后才能删除其它的主机。删除主机时不能打开该主机上的 osd 目录。





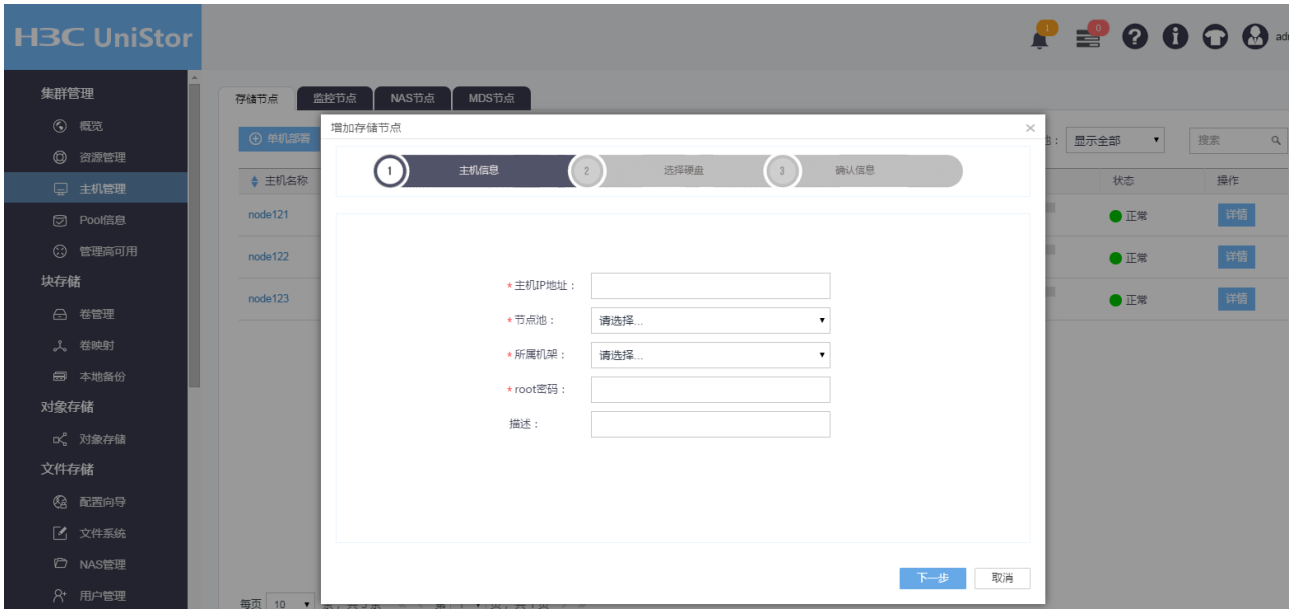
注：针对不能打开 ONEStor 版本的情况，请执行如下命令临时打开(主备都需执行)：

mv /opt/h3c/webapp/content/dsm/index.html.bak /opt/h3c/webapp/content/dsm/index.html

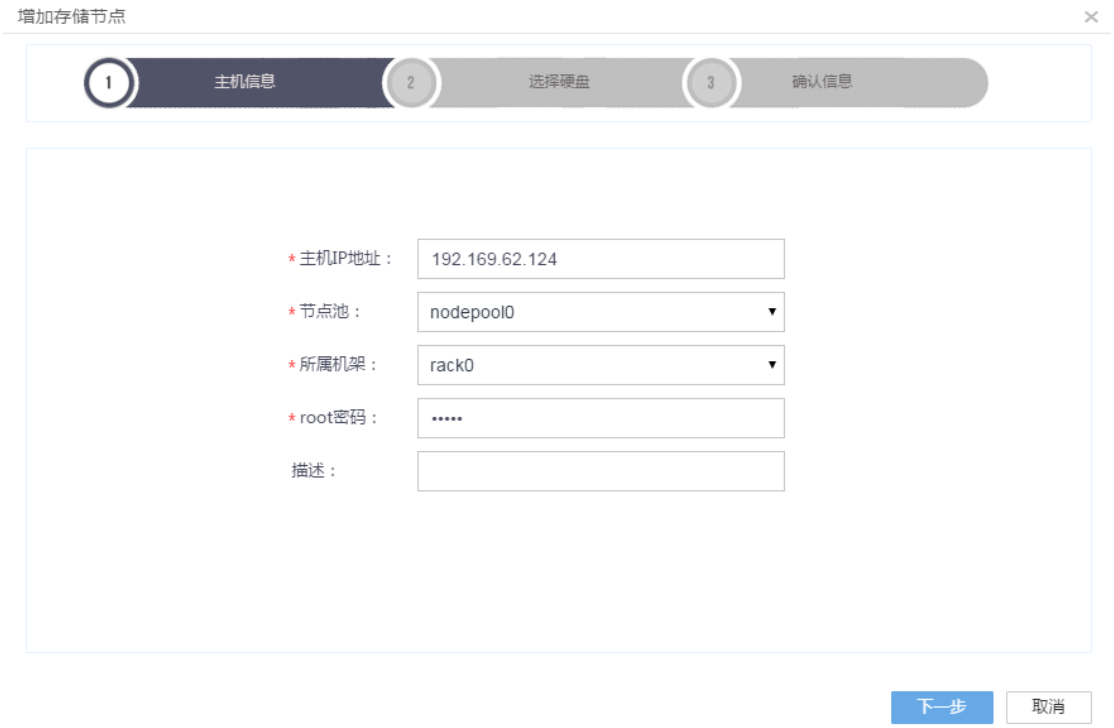
3.14.7 在 ONEStor 界面进行扩容添加主机

在 ONEStor 管理界面添加该服务器，使用单机部署方式添加主机

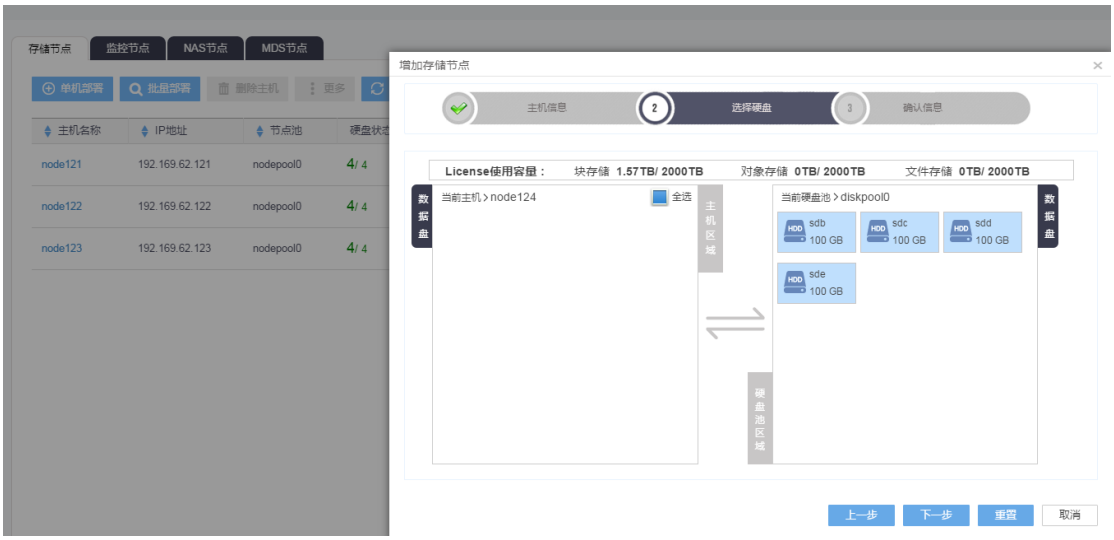
选择 集群管理==》主机管理==》存储节点==》单机部署 后如下图



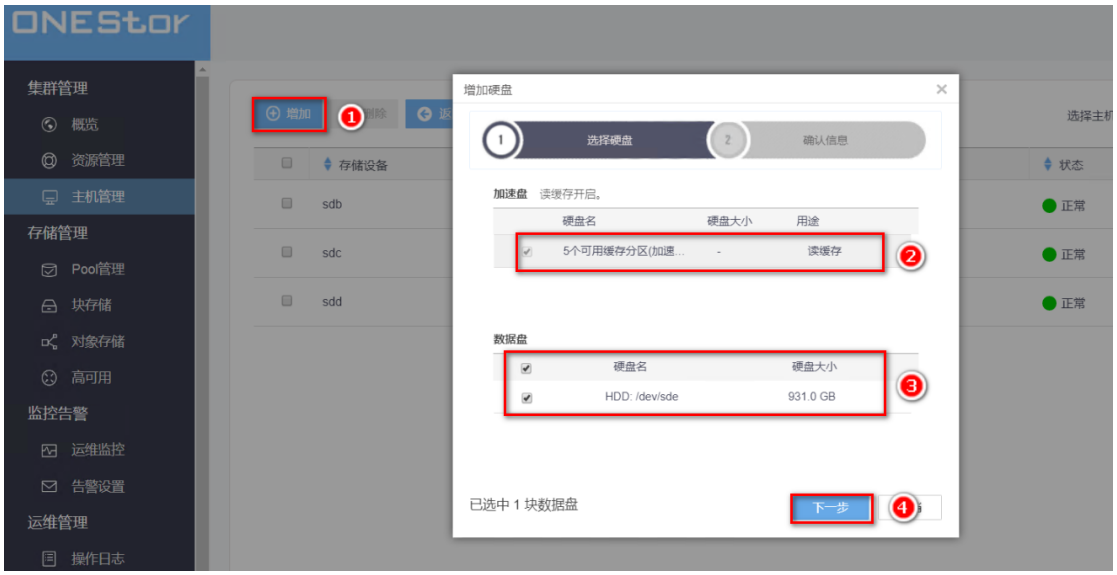
填入需要添加的服务器的管理网 IP，选择节点池和机架并输入 root 用户密码后点击下一步（此处注意，节点池和机架应与原集群的机架一致）



根据硬盘池配置规则，选择数据盘加入对应的硬盘池（此处注意：主机扩容时：需满足硬盘池下各主机间加入的数据盘数相差不大于 1 的限制），点击下一步



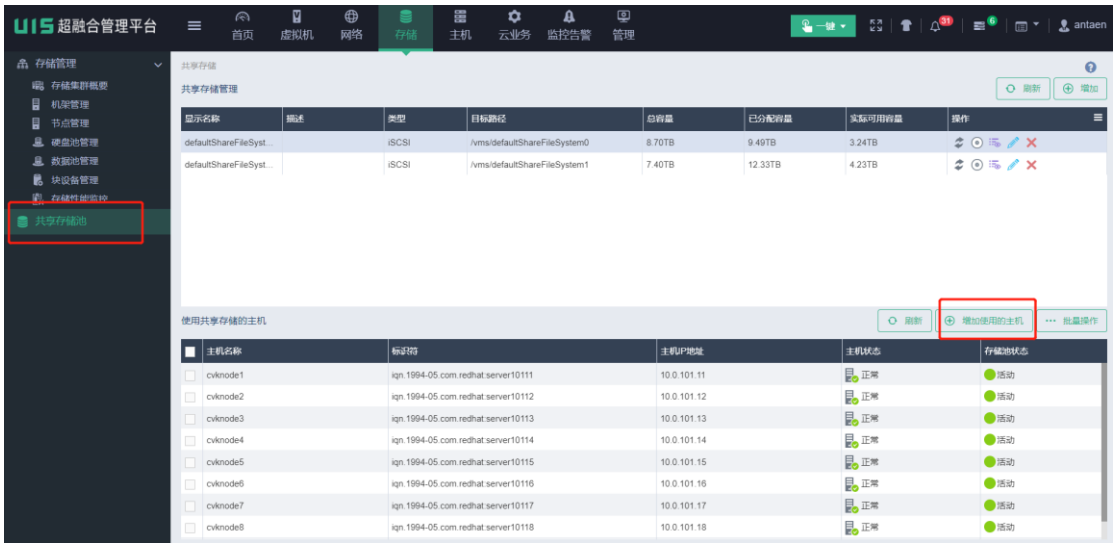
之后确认配置信息，信息无误则点击确定

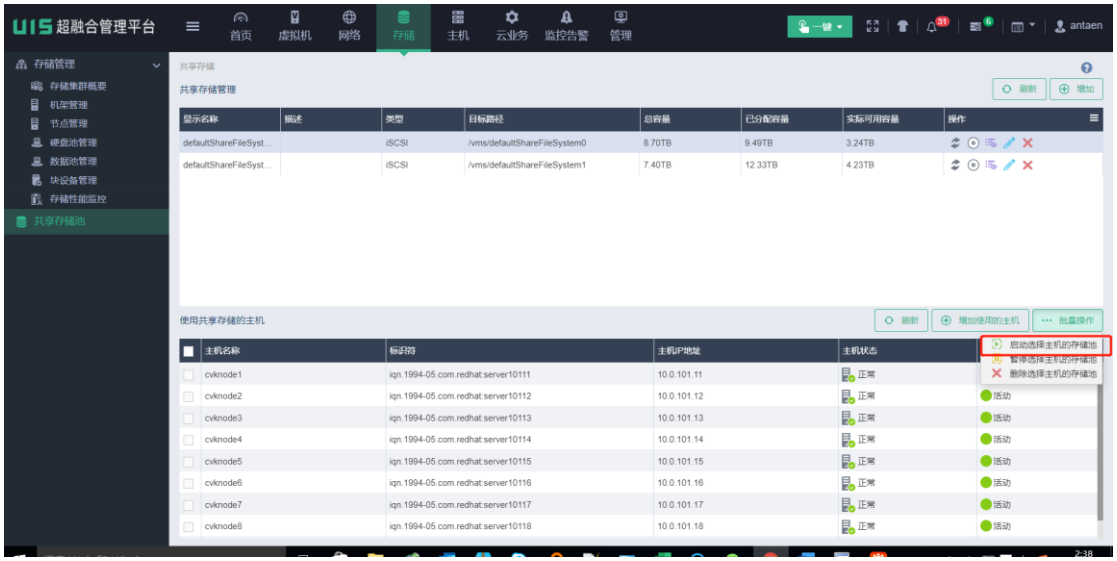


3.14.8 等待 ONEStor 数据平衡

等待 ONEStor 数据完全平衡，集群健康度为 100%，且无任何告警。

3.14.9 添加并启动共享存储





3.14.10 在每个节点变更缓存分区

确保 onestor 数据健康度为 100%，在每个节点上依次重复执行步骤 3.14.4-3.14.8，完成主机从 onestor 集群删除再添加。进行缓存变更操作。所有节点更换完成后还原副本数。

注意：在线扩容、缩容会引起较多的数据均衡，建议在业务量低时实施；

4 日志收集和介绍

4.1 UIS系统日志

4.1.1 UIS 日志收集

1. UIS 页面收集

在[管理/日志文件收集]页面收集 UIS 系统的日志文件。

选择需要收集的 CVK 主机，并点击<收集日志文件>按钮，将日志文件保存到本地电脑。

2. CVK 后台手工收集

如果 CVK 异常，无法从 UIS 平台页面收集日志文件，则可以通过登录 CVK 主机后台进行手工收集。在 CVK 主机后台执行“cas_collect_log.sh”命令，收集完成后在“/vms”目录下会生产该 CVK 主机的日志文件，如下图所示。

可以通过 SSH 客户端软件下载到本地电脑进行分析。

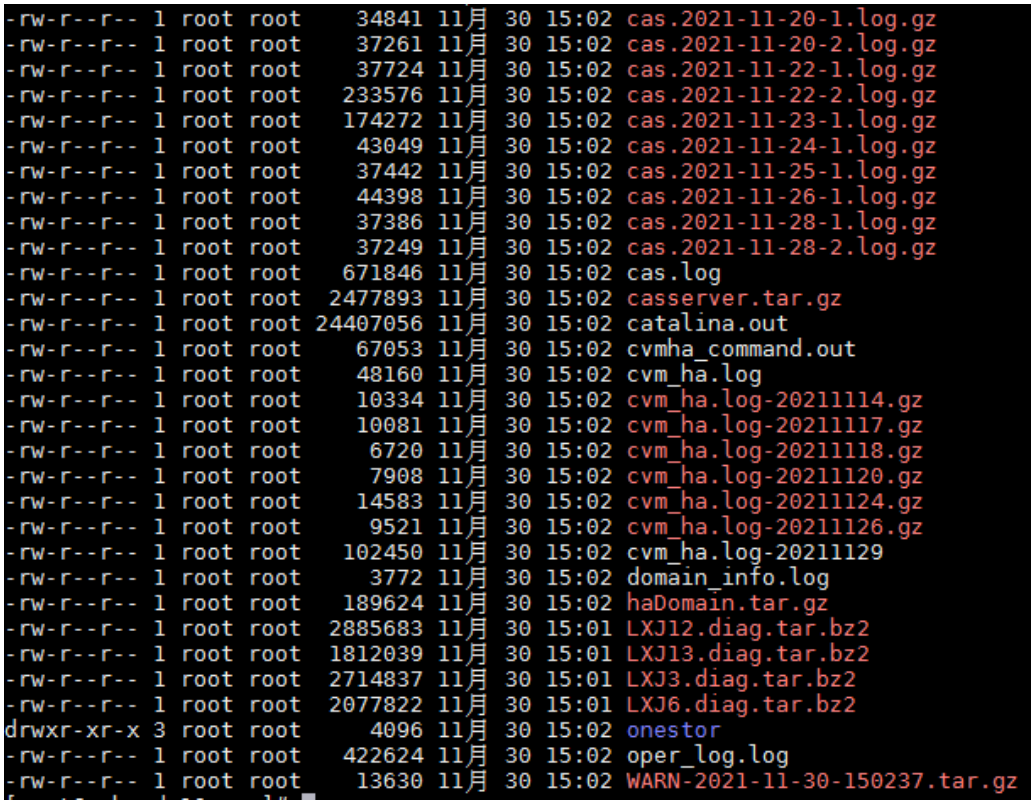
```
root@cvknode1:~# cas_collect_log.sh -h

SHELL NAME: cas collect log.sh
USAGE     :cas_collect_log.sh $time $size
PARAMETER :
     time : collect log last time days
     size : size of logs(KB)

root@cvknode1:~# cas collect log.sh 1 10000
  No volume groups found
cp: cannot stat `/var/lib/heartbeat/crm/*': No such file or directory
cp: cannot stat `/var/lib/libvirt/qemu/snapshot/*': No such file or directory
find: `/var/log/upgrade': No such file or directory
find: `/var/log/upgrade': No such file or directory
cp: cannot stat `/var/log/cvm-upgrade.log': No such file or directory
cp: cannot stat `/var/log/castools.log': No such file or directory
cp: cannot stat `/var/log/ocfs2_fence_restart.log': No such file or directory
root@cvknode1:~# ll /vms/cvknode1.diag.tar.bz2
-rw-r--r-- 1 root root 2574416 Apr 21 10:18 /vms/cvknode1.diag.tar.bz2
root@cvknode1:~#
```

ONEStor 相关无法执行脚本收集，需要手动拷贝/var/log/storage, /var/log/ceph 日志，如果所需要的日志周期较短，或者以上日志太大，可以只收集归档/var/log/storage/backup 的一部分。

4.1.2 日志介绍

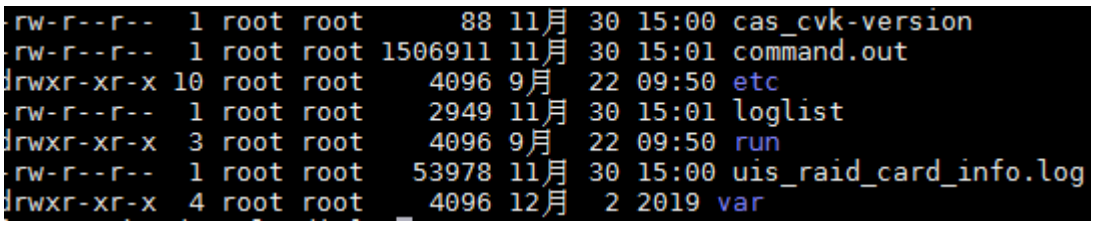
下载的 UIS 日志文件名为“UIS_×××_×××.tar.gz”。

解压缩日志文件后主要包含如下几种文件：



- catalina.out：UIS WEB 功能日志
- oper_log.log：用户的操作日志
- *.diag.tar.bz2 各 CVK 主机日志
- onestor 分布式存储的日志，包括操作日志和系统日志
- WARN*.tar.gz 告警信息

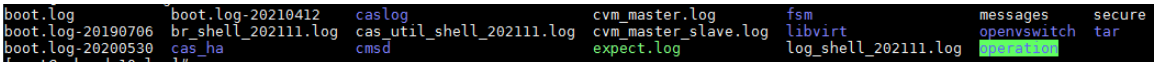
解压缩 CVK 主机的“XXX.tar.bz2”日志文件，加压缩后包含了如下的目录文件：



- etc:目录包含了 UIS 配置文件，最主要为虚拟机的配置文件，路径为“libvirt/qemu/VM.xml”
- var：目录包含了 UIS 各个功能模块的日志信息。
- command.out：后台常用命令输出信息。
- cas _cvk-version：UIS 版本信息。
- loglist：UIS 的 log 日志文件名信息。

- uis_raid_card_info.log：主机 raid 卡的基本信息。

Var 目录包含了 UIS 各个功能模块的日志信息，主要日志包括如下：



- messages：主机系统日志记录系统运行信息
- fsm：共享文件系统日志
- cas_ha：HA 日志（自研）
- Ha_shell_XX.log：HA 日志
- Libvirt：虚拟机相关日志
- Openvswtich：虚拟交换机日志
- Ovs_shell_XX.log：虚拟交换机日志
- Tomcat8：UIS WEB 日志
- Operation：UIS 后台手工执行的日志

CVK 主机日志说明：

(1) messages 日志介绍

messages 日志记录了操作系统运行中重要的信息，如下介绍 CVK 主机常见问题的信息记录。

(2) CVK 主机异常重启

如下信息所示，在 13:58:01 和 14:06:35 之间 messages 日志文件中没有任何的信息记录，说明该时间段内 CVK 主机异常。

后面 Kernel 级别的日志记录 CVK 主机发生重启后的信息。

```
Feb 3 13:58:01 XJYZ-CVK01 CRON【64458】: (root) CMD (ump-node-sync )
Feb 3 13:58:01 XJYZ-CVK01 CRON【64459】: (root) CMD (ump-sync -p ALL)
Feb 3 13:58:01 XJYZ-CVK01 CRON 【 64460 】 : (root) CMD ( /opt/bin/ocfs2_iscsi_conf_chg_timer.sh)
Feb 3 13:58:01 XJYZ-CVK01 CRON【64443】: (CRON) info (No MTA installed, discarding output)
Feb 3 14:06:35 XJYZ-CVK01 kernel: imklog 5.8.6, log source = /proc/kmsg started.
Feb 3 14:06:35 XJYZ-CVK01 rsyslogd: 【 origin software="rsyslogd" swVersion="5.8.6" x-pid="2747" x-info="http://www.rsyslog.com"】 start
Feb 3 14:06:35 XJYZ-CVK01 rsyslogd: rsyslogd's groupid changed to 103
Feb 3 14:06:35 XJYZ-CVK01 rsyslogd: rsyslogd's userid changed to 101
Feb 3 14:06:35 XJYZ-CVK01 rsyslogd-2039: Could not open output pipe '/dev/xconsole' 【try http://www.rsyslog.com/e/2039 】
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【0.000000】 Initializing cgroup subsys cpuset
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【0.000000】 Initializing cgroup subsys cpu
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【0.000000】 Initializing cgroup subsys cpuacct
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【0.000000】 Linux version 3.13.6 (root@cvknode22) (gcc version 4.6.3 (Ubuntu/Linaro 4.6.3-1ubuntu5) ) #5 SMP Mon Jul 21 10:07:26 CST 2014
```

```
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【0.000000】 Command line: BOOT_IMAGE=/boot/vmlinuz-3.13.6
root=UUID=4beeb503-6e10-4836-93a4-0836a9a1571e ro nomodeset elevator=deadline
transparent_hugepage=always crashkernel=256M quiet
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【0.000000】 KERNEL supported cpus:
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【0.000000】 Intel GenuineIntel
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【0.000000】 AMD AuthenticAMD
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【0.000000】 Centaur CentaurHauls
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【0.000000】 e820: BIOS-provided physical RAM map:
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【 0.000000 】 BIOS-e820: 【 mem
0x0000000000000000-0x000000000009cbff】 usable
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【 0.000000 】 BIOS-e820: 【 mem
0x000000000009cc00-0x000000000009ffff】 reserved
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【 0.000000 】 BIOS-e820: 【 mem
0x00000000000f0000-0x00000000000fffff】 reserved
Feb 3 14:06:35 XJYZ-CVK01 kernel: 【 0.000000 】 BIOS-e820: 【 mem
0x0000000000100000-0x00000000bf60ffff】 usable
```

(3) Libvirt 日志介绍

如下所示，日志文件【/var/log/libvirt/libvirtd.log】，提示 CVK 主机的缺少内存资源告警，当前内存占用率已经高达 97%。（CPU 资源不足时提示信息类似）

```
2014-10-24 09:15:52.792+0000: 2994: warning : virIsLackOfResource:1106 : Lack of Memory
resource! only 374164 free 64068 cached and vm locked memory(4194304*0%) of 16129760 total,
max:85; now:97
2014-10-24 09:15:52.792+0000: 2994: error : qemuProcessStart:3419 : Lack of system resources,
out of memory or cpu is too busy, please check it.
```

日志目录【/var/log/libvirt/qemu】保存了运行在该 CVK 主机上的虚拟机日志文件，如下所示。

```
root@UIS-CVK01:/var/log/libvirt/qemu# ls -l
total 44
-rw------- 1 root root  7067 Jan  9 19:08 RedHat5.9.log
-rw------- 1 root root  1969 Jan 18 15:41 win7.log
-rw------- 1 root root 26574 Feb 11 16:15 windows2008.log
```

虚拟机日志文件记录了虚拟机运行信息，如虚拟机的启动时间、关闭时间、虚拟机的磁盘文件等信息。

```
2015-02-11 15:50:18.349+0000: starting up
LC_ALL=C PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
QEMU_AUDIO_DRV=none /usr/bin/kvm -name windows2008 -S -machine
pc-i440fx-1.5,accel=kvm,usb=off,system=windows -cpu qemu64,hv_relaxed,hv_spinlocks=0x2000
-m 1024 -smp 1,maxcpus=12,sockets=12,cores=1,threads=1 -uuid
43741f06-166d-4155-b47e-4137df68e91c -no-user-config -nodefaults -chardev
file=/vms/sharefile/windows2008,if=none,id=drive-virtio-disk0,format=qcow2,cache=directs
ync -device
```

```
…..
char device redirected to /dev/pts/0 (label charserial0)
qemu: terminating on signal 15 from pid 4530
2015-02-11 16:15:28.825+0000: shutting down
```

(4) OCFS2 日志介绍

如下所示，日志文件【/var/log/fsm/fsm_core*.log】会记录 CVK 主机由于 ocfs2 fence 触发处理的信息。

```
2021-11-04 06:40:35,882 manager:233 INFO Received an event: {'index': 7, 'type': 'fence_umount', 'uuid': u'851D36905AB74AFD93E1ABA8259DA3A2', 'seq': 11538, 'dev_name': u'dm-7'}
2021-11-04 06:40:35,923 manager:204 INFO Remain 0 events to be handling
2021-11-04 06:40:35,923 manager:131 INFO Manager received an event: Pool sharefile06 was fence_umount
2021-11-04 06:40:35,923 fspool:141 INFO Pool sharefile06 received a event fence_umount
```

(5) Operation 日志介绍

Operation 日志记录了在 CVK 后台执行的命令信息。如下所示包含 4 月 19 到 4 月 21 日三天的信息。

```
root@cvknode1:~/cas# ll /var/log/operation/
total 32
drwxrwxrwx  2 root root 4096 Apr 21 10:06 ./
drwxr-xr-x 40 root root 4096 Apr 21 11:01 ../
-rwxrwxrwx  1 root root 5162 Apr 19 17:49 18-04-19.log*
-rwxrwxrwx  1 root root  829 Apr 20 19:11 18-04-20.log*
-rwxrwxrwx  1 root root 8505 Apr 21 11:00 18-04-21.log*
```

Operation 日志文件的信息内容如下所示，包括了命令的执行时间、登录用户、登录地址和登录方式、以及具体命令和执行命令时所在的目录信息。

```
2018/04/19 16:56:50##root pts/6 (172.16.130.3)##/root## vi /var/log/tomcat8/cas.log
2018/04/19 16:57:05##root pts/6 (172.16.130.3)##/root## service tomcat8 restart
2018/04/19 17:02:21##root pts/5 (172.16.130.3)##/root## cat /etc/cvk/system_alarm.xml
2018/04/19 17:02:23##root pts/5 (172.16.130.3)##/root## lsblk
2018/04/19 17:49:04##root pts/6 (172.16.130.3)##/root## ceph osd tree
2018/04/19 17:49:19##root pts/6 (172.16.130.3)##/root## stop ceph-osd id=3
```

4.2 虚拟机的castools工具日志

UIS 系统和虚拟机是相互隔离的，为了实现 UIS 系统对虚拟机的监控和管理，需要在虚拟机内部的操作系统中安装 castools 工具。

根据虚拟机操作系统的不同，castools 工具的日志收集方法也分为两种方法：

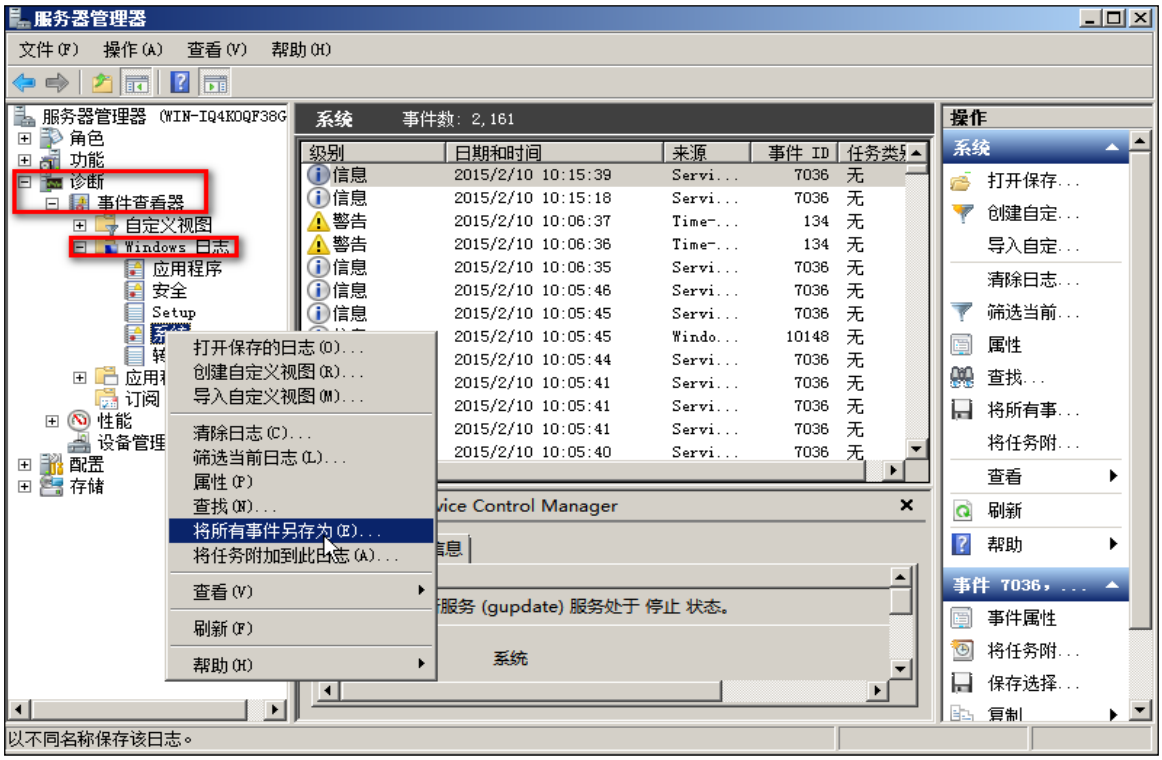
- Windows 虚拟机：虚拟机内部获取文件“C:\Program Files\castools\qemu-ga.log”
- Linux 虚拟机：虚拟机内部获取文件“/var/log/qemu-ga.log”和“/var/log/set-ip.log”

4.3 虚拟机操作系统日志

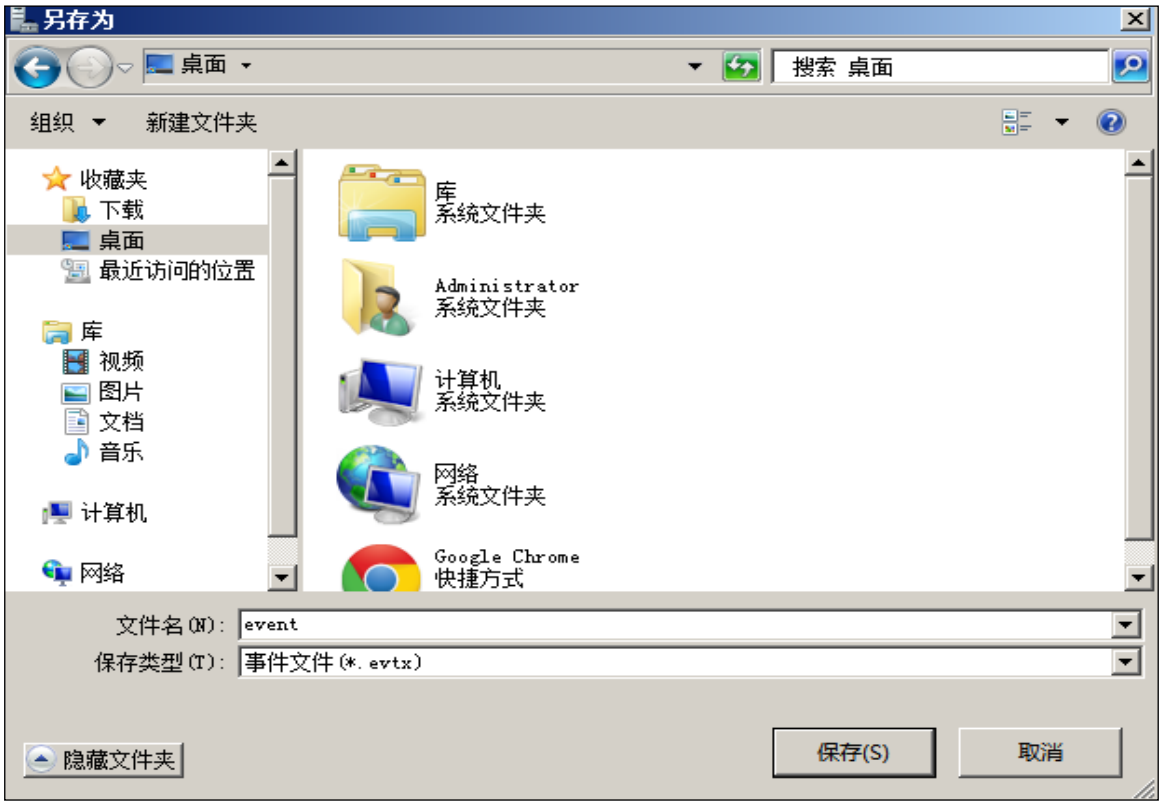
4.3.1 Windows 操作系统日志收集

在【服务器管理器】对话框中，进入“【诊断】/【事件查看器】/【Windows 日志】”页面收集 Windows 的系统日志和应用程序日志。

收集方法如下图所示，右键点击【系统】按钮，选择“将所有事件另存为”。



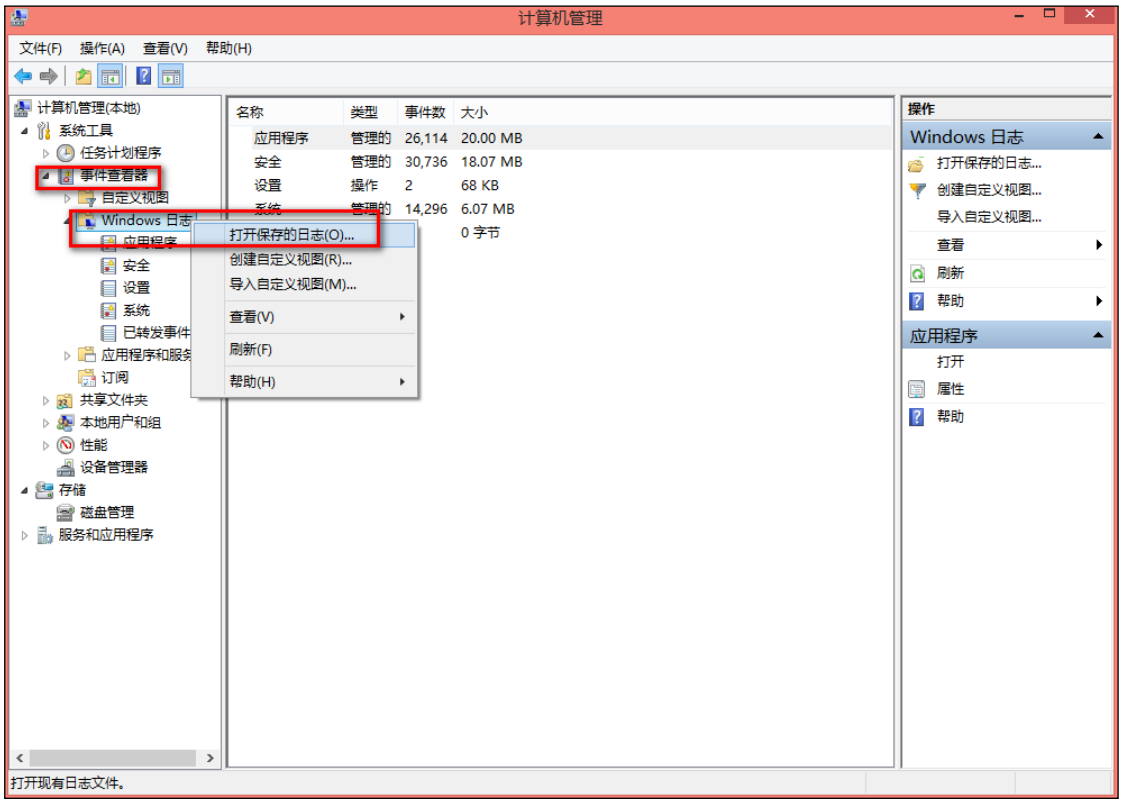
将日志进行保存。



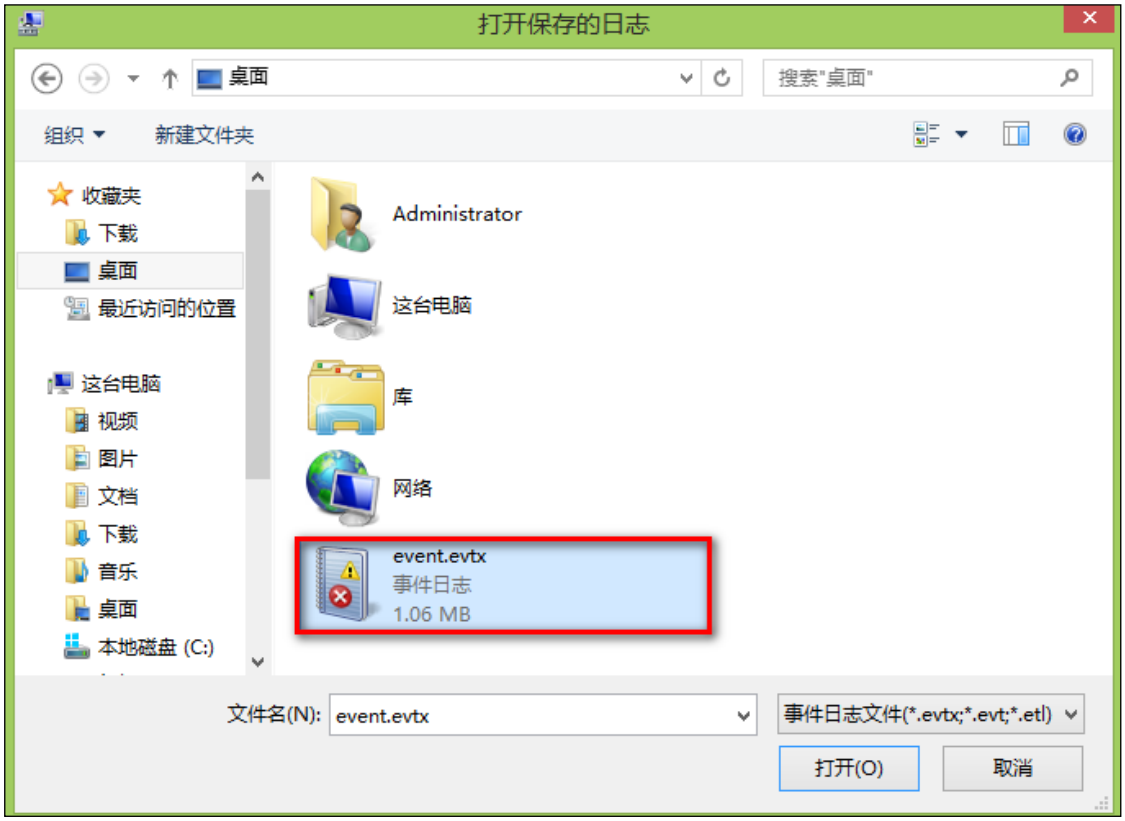
下载完成后的日志如下图所示。

4.3.2 Windows 操作系统日志查看

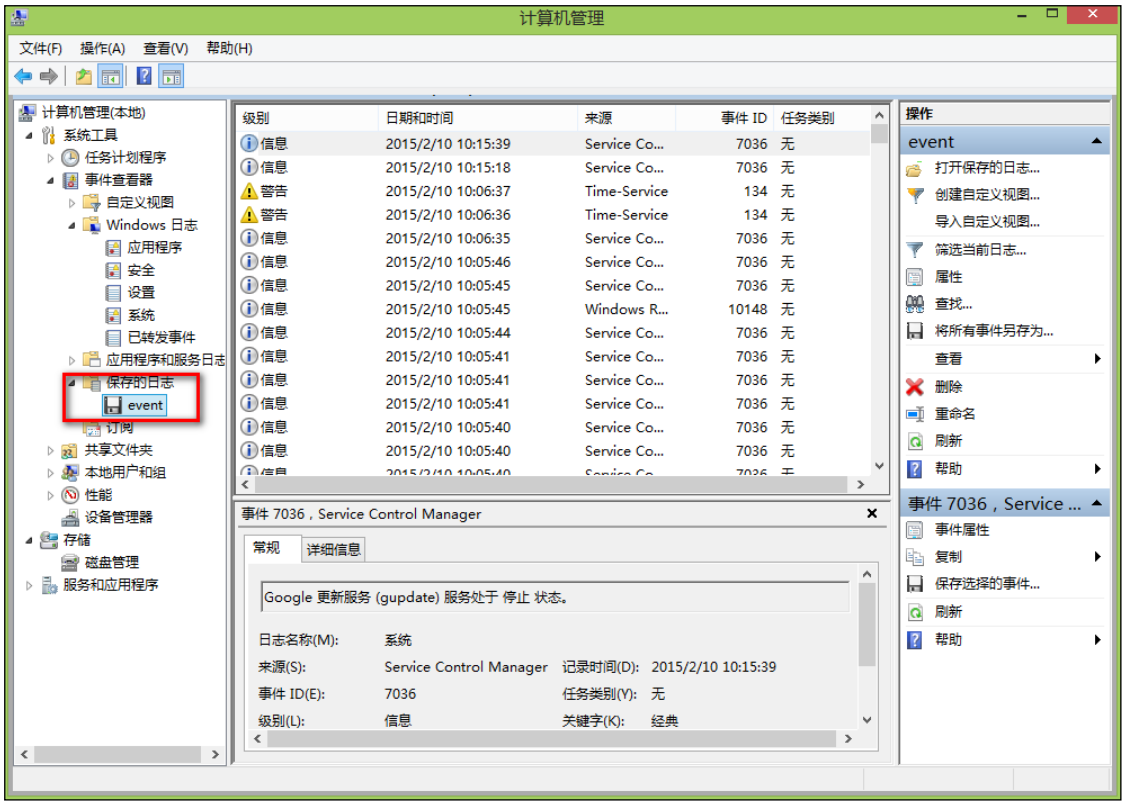
在本地电脑获取到虚拟机的日志文件后，在本地电脑（Windows7）打开【计算机管理】对话框中的【事件查看器】页面右键点击【Windows 日志】按钮，选择“打开保存的日志”。



在弹出的【打开保存的日志】对话框中选择日志。



打开完成后，在【保存的日志】页面显示打开的日志。



4.3.3 Linux 操作系统日志收集

针对 Linux 操作系统的虚拟机，需要收集/var/log 下的日志即可，针对日志较大的情况，需要进行压缩然后再进行拷贝出来。例如 2019-09-17 收集 vm_test 虚拟机的日志：

tar -cvf vm_test_20190917.tar.gz /var/log

4.4 UIS主机异常问题定位工具使用介绍

4.4.1 Kdump 介绍

Kdump 是 Linux 内核的一个转储工具，其基本原理是在内存中保留一块区域，这块区域用来存放 capture kernel，当前的内核发生 crash 后，通过 kexec 把保留区域的 capture kernel 运行起来，由 capture kernel 负责把 crash kernel 的完整信息，包括 CPU 寄存器、堆栈数据等现场重要数据转储到文件中，文件的存放位置可以是本地磁盘，也可以是网络。

UIS 系统默认支持 Kdump 功能，在 CVK 主机的内核异常时，会在/vms/crash 目录下生成 crash 文件，以方便后续问题定位，比如某 CVK 在一次异常时生成的 crash 文件如下：

```
root@cvk29:/vms/crash# ls -lt
```

```
drwxr-sr-x 2 root whoopsie      4096 Jul 22 17:34 2014-07-22-09:34
```

目录“2014-07-22-09:34”中有一个 dump-***的文件就是 kdump 的输出。

4.4.2 Kdump 文件分析

可以使用 crash 工具分析内核 Kdump 转储文件。在分析时，需要用到内核版本的 vmlinux 文件，这个文件放在/usr/src/linux-4.1.0-generic/vmlinux-×××（不同的内核版本名称略有不同）。

下面以几个典型的网上问题来说明下。

1. CPU 故障

某局点反映 cvknode1 节点反复重启，将节点上的虚拟机全部迁移走，并删除了共享存储配置后，仍然反复重启，查看重启时刻的 syslog 信息，重启前没有任何异常信息，且存在/vms/crash 下存在 vmcore 文件。

异常调用栈：

```
root@cvk21:/vms/tmp# crach vmlinux vmcore
No command 'crach' found, did you mean:
Command 'crash' from package 'crash' (main)
crach: command not found
root@cvk21:/vms/tmp# crash vmlinux vmcore

crash 7.0.5
Copyright (C) 2002-2014  Red Hat, Inc.
Copyright (C) 2004, 2005, 2006, 2010  IBM Corporation
Copyright (C) 1999-2006  Hewlett-Packard Co
Copyright (C) 2005, 2006, 2011, 2012  Fujitsu Limited
Copyright (C) 2006, 2007  VA Linux Systems Japan K.K.
Copyright (C) 2005, 2011  NEC Corporation
Copyright (C) 1999, 2002, 2007  Silicon Graphics, Inc.
Copyright (C) 1999, 2000, 2001, 2002  Mission Critical Linux, Inc.
This program is free software, covered by the GNU General Public License,
and you are welcome to change it and/or distribute copies of it under
certain conditions.  Enter "help copying" to see the conditions.
This program has absolutely no warranty.  Enter "help warranty" for details.

GNU gdb (GDB) 7.6
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later 【http://gnu.org/licenses/gpl.html】
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-unknown-linux-gnu"...

      KERNEL: vmlinux
```

```
    DUMPFILE: vmcore  【PARTIAL DUMP】
        CPUS: 8
        DATE: Wed Nov  5 12:25:19 2014
      UPTIME: 00:02:19
LOAD AVERAGE: 0.06, 0.05, 0.02
       TASKS: 324
    NODENAME: cvknode-1
     RELEASE: 3.13.6
     VERSION: #5 SMP Mon Jul 21 10:07:26 CST 2014
     MACHINE: x86_64  (2132 Mhz)
      MEMORY: 64 GB
       PANIC: "Kernel panic - not syncing: Fatal Machine check"
         PID: 0
     COMMAND: "swapper/6"
        TASK: ffff8807f4618000  (1 of 8)  【THREAD_INFO: ffff8807f4620000】
         CPU: 6
       STATE: TASK_RUNNING (PANIC)

crash】 bt
PID: 0      TASK: ffff8807f4618000  CPU: 6   COMMAND: "swapper/6"
#0 【ffff8807ffc6ac50】 machine_kexec at ffffffff8104c991
#1 【ffff8807ffc6acc0】 crash_kexec at ffffffff810e97e8
#2 【ffff8807ffc6ad90】 panic at ffffffff8174ac9d
#3 【ffff8807ffc6ae10】 mce_panic at ffffffff81038b2f
#4 【ffff8807ffc6ae60】 do_machine_check at ffffffff810399d8
#5 【ffff8807ffc6af50】 machine_check at ffffffff817589df
    【exception RIP: intel_idle+204】
    RIP: ffffffff8141006c  RSP: ffff8807f4621db8  RFLAGS: 00000046
    RAX: 0000000000000010  RBX: 0000000000000004  RCX: 0000000000000001
    RDX: 0000000000000000  RSI: ffff8807f4621fd8  RDI: 0000000001c0d000
    RBP: ffff8807f4621de8   R8: 0000000000000009   R9: 0000000000000004
    R10: 0000000000000001  R11: 0000000000000001  R12: 0000000000000003
    R13: 0000000000000010  R14: 0000000000000002  R15: 0000000000000003
    ORIG_RAX: ffffffffffffffff  CS: 0010  SS: 0018
--- 【MCE exception stack】 ---
#6 【ffff8807f4621db8】 intel_idle at ffffffff8141006c
#7 【ffff8807f4621df0】 cpuidle_enter_state at ffffffff81602a8f
#8 【ffff8807f4621e50】 cpuidle_idle_call at ffffffff81602be0
```

```
#9 【ffff8807f4621ea0】 arch_cpu_idle at ffffffff8101e2ce
#10 【ffff8807f4621eb0】 cpu_startup_entry at ffffffff810c1818
#11 【ffff8807f4621f20】 start_secondary at ffffffff8104306b
crash】
```

从异常栈可以看到，出现 MCE exception，即 Machine Check Error，出现这样的异常情况一般是硬件问题。

异常前的 dmesg 信息：

crash-dmesg

从 dmesg 输出的信息中，可以看到在异常重启前出现了如下的打印信息：

```
【   15.707981】 8021q: 802.1Q VLAN Support v1.8
【   16.416569】 drbd: initialized. Version: 8.4.3 (api:1/proto:86-101)
【   16.416573】 drbd: srcversion: F97798065516C94BE0F27DC
【   16.416575】 drbd: registered as block device major 147
【   17.142281】 Ebtables v2.0 registered
【   17.203400】 ip_tables: (C) 2000-2006 Netfilter Core Team
【   17.247387】 ip6_tables: (C) 2000-2006 Netfilter Core Team
【  139.114172】 Disabling lock debugging due to kernel taint
【   139.114185】 mce: 【Hardware Error】: CPU 2: Machine Check Exception: 4 Bank 5:
be00000000800400
【  139.114192】 mce: 【Hardware Error】: TSC 10ba0482e78 ADDR 3fff81760d32 MISC 7fff
【  139.114199】 mce: 【Hardware Error】: PROCESSOR 0:206c2 TIME 1415161519 SOCKET 0 APIC 14
microcode 13
【  139.114203】 mce: 【Hardware Error】: Run the above through 'mcelog --ascii'
【  139.114208】 mce: 【Hardware Error】: Machine check: Processor context corrupt
【  139.114211】 Kernel panic - not syncing: Fatal Machine check
crash】
```

从以上的信息基本可以确定是硬件 CPU2 存在问题导致的。

2. 内存故障

某局点反映 cvk 节点无故重启，分析 syslog 在重启前后的日志信息，没有发现异常记录。

异常调用栈：

重启时，存在 kdump 记录，查看调用栈如下，初步判断应该是硬件问题。

```
crash】 bt
PID: 0      TASK: ffffffff81c144a0  CPU: 0   COMMAND: "swapper/0"
#0 【ffff880c0fa07c60】 machine_kexec at ffffffff8104c991
#1 【ffff880c0fa07cd0】 crash_kexec at ffffffff810e97e8
#2 【ffff880c0fa07da0】 panic at ffffffff8174ac9d
#3 【ffff880c0fa07e20】 asminline_call at ffffffffa014c895 【hpwdt】
#4 【ffff880c0fa07e40】 nmi_handle at ffffffff817598da
```

```
#5 【ffff880c0fa07ec0】 do_nmi at ffffffff81759b7d
#6 【ffff880c0fa07ef0】 end_repeat_nmi at ffffffff81758cf1
    【exception RIP: intel_idle+204】
    RIP: ffffffff8141006c  RSP: ffffffff81c01da8  RFLAGS: 00000046
    RAX: 0000000000000010  RBX: 0000000000000010  RCX: 0000000000000046
    RDX: ffffffff81c01da8  RSI: 0000000000000018  RDI: 0000000000000001
    RBP: ffffffff8141006c   R8: ffffffff8141006c   R9: 0000000000000018
    R10: ffffffff81c01da8  R11: 0000000000000046  R12: ffffffffffffffff
    R13: 0000000000000000  R14: ffffffff81c01fd8  R15: 0000000000000000
    ORIG_RAX: 0000000000000000  CS: 0010  SS: 0018
--- 【NMI exception stack】 ---
#7 【ffffffff81c01da8】 intel_idle at ffffffff8141006c
#8 【ffffffff81c01de0】 cpuidle_enter_state at ffffffff81602a8f
#9 【ffffffff81c01e40】 cpuidle_idle_call at ffffffff81602be0
#10 【ffffffff81c01e90】 arch_cpu_idle at ffffffff8101e2ce
#11 【ffffffff81c01ea0】 cpu_startup_entry at ffffffff810c1818
#12 【ffffffff81c01f10】 rest_init at ffffffff8173fc97
#13 【ffffffff81c01f20】 start_kernel at ffffffff81d37f7b
#14 【ffffffff81c01f70】 x86_64_start_reservations at ffffffff81d375f8
#15 【ffffffff81c01f80】 x86_64_start_kernel at ffffffff81d3773e
crash】
```

异常前的 dmesg 信息：

```
crash】dmesg
……
【10753.155822】 sd 3:0:0:1: 【sdd】 Very big device. Trying to use READ CAPACITY(16).
【10804.115376】 sbridge: HANDLING MCE MEMORY ERROR
【10804.115386】 CPU 23: Machine Check Exception: 0 Bank 9: cc1bc010000800c0
【10804.115387】 TSC 0 ADDR 12422f7000 MISC 90868002800208c PROCESSOR 0:306e4 TIME 1417366012
SOCKET 1 APIC 2b
……
【10804.283467】 sbridge: HANDLING MCE MEMORY ERROR
【10804.283473】 CPU 9: Machine Check Exception: 0 Bank 9: cc003010000800c0
【10804.283475】 TSC 0 ADDR 1242ef7000 MISC 90868000800208c PROCESSOR 0:306e4 TIME 1417366012
SOCKET 1 APIC 26
【10804.303482】 EDAC MC1: 28416 CE memory scrubbing error on CPU_SrcID#1_Channel#0_DIMM#0
(channel:0 slot:0 page:0x12422f7 offset:0x0 grain:32 syndrome:0x0 -  OVERFLOW area:DRAM
err_code:0008:00c0 socket:1 channel_mask:1 rank:0)
【10804.303489】 EDAC MC1: 192 CE memory scrubbing error on CPU_SrcID#1_Channel#0_DIMM#0
(channel:0 slot:0 page:0x12424a7 offset:0x0 grain:32
```

……

【10804.319474】 sbridge: HANDLING MCE MEMORY ERROR

【10804.319481】 CPU 6: Machine Check Exception: 0 Bank 9: cc001010000800c0

【10804.319482】 TSC 0 ADDR 1243087000 MISC 90868002800208c PROCESSOR 0:306e4 TIME 1417366012 SOCKET 1 APIC 20

【10805.303772】 EDAC MC1: 64 CE memory scrubbing error on CPU_SrcID#1_Channel#0_DIMM#0 (channel:0 slot:0 page:0x1243087 offset:0x0 grain:32 syndrome:0x0 - OVERFLOW area:DRAM err_code:0008:00c0 socket:1 channel_mask:1 rank:0)

【10813.602696】 sd 3:0:0:0: 【sdc】 Very big device. Trying to use READ CAPACITY(16).

【10813.603219】 sd 3:0:0:1: 【sdd】 Very big device. Trying to use READ CAPACITY(16).

【10840.833238】 Kernel panic - not syncing: An NMI occurred, please see the Integrated Management Log for details.

crash】

kern. log 信息：

syslog 中虽然没有记录，但是在 kern.log 中可以看到类似如下的日志信息：

Nov 30 07:05:01 HBND-UIS-E-CVK09 kernel: 【229821.496666】 sd 11:0:0:1: 【sdd】 Very big device. Trying to use READ CAPACITY(16).

Nov 30 07:05:55 HBND-UIS-E-CVK09 kernel: 【229875.188854】 sbridge: HANDLING MCE MEMORY ERROR

Nov 30 07:05:55 HBND-UIS-E-CVK09 kernel: 【229875.188873】 CPU 23: Machine Check Exception: 0 Bank 9: cc1e0010000800c0

Nov 30 07:05:55 HBND-UIS-E-CVK09 kernel: 【229875.188874】 TSC 0 ADDR 10638f7000 MISC 90868002800208c PROCESSOR 0:306e4 TIME 1417302355 SOCKET 1 APIC 2b

……

Nov 30 07:05:55 HBND-UIS-E-CVK09 kernel: 【229875.244902】 EDAC MC1: 30720 CE memory scrubbing error on CPU_SrcID#1_Channel#0_DIMM#0 (channel:0 slot:0 page:0x10638f7 offset:0x0 grain:32 syndrome:0x0 - OVERFLOW area:DRAM err_code:0008:00c0 socket:1 channel_mask:1 rank:0)

……

root@gzh-139:/vms/issue_logs/hebeinongda/20141201/HBND-UIS-E-CVK09/logdir/var/log# grep OVERFLOW kern* | wc

225 6341 60264

root@gzh-139:/vms/issue_logs/hebeinongda/20141201/HBND-UIS-E-CVK09/logdir/var/log#

从以上的信息基本可以确定是内存有问题导致的。现场更换内核后问题解决。

/var/log/ceph/ceph. log

ceph.log 主要记录集群的健康状况以及集群的流量等内容，只有监控节点才有，内容与 ceph -w 查看内容一致；

- 若是在 ceph 日志中发现打印如下异常日志，原因是集群主 moniter 节点业务网断开；

```
2017-05-09 19:44:03.400143 mon.2 172.16.105.84:6789/0 2009 : cluster [INF] mon.cvknode84 calling new monitor election
2017-05-09 19:44:03.404362 mon.1 172.16.105.83:6789/0 2023 : cluster [INF] mon.cvknode83 calling new monitor election
2017-05-09 19:44:05.419510 mon.1 172.16.105.83:6789/0 2024 : cluster [INF] mon.cvknode83@1 won leader election with quorum 1,2
2017-05-09 19:44:05.428131 mon.1 172.16.105.83:6789/0 2025 : cluster [INF] HEALTH_WARN; 1 mons down, quorum 1,2 cvknode83,cvknode84
2017-05-09 19:44:14.383590 mon.1 172.16.105.83:6789/0 2057 : cluster [INF] osdmap e1397: 18 osds: 12 up, 18 in
```

- 若是在 ceph 日志中发现打印如下异常日志，原因是集群健康度不为 100%，集群正处于恢复状态；

```
2017-06-06 19:31:41.319993 mon.0 192.168.93.21:6789/0 86387 : cluster [INF] pgmap v73931: 4096 pgs: 2561 active+clean, 1532 active+remapped+wait_backfill, 3 active+remapped+backfilling; 3362 GB data, 6730 GB used, 21941 GB / 28672 GB avail; 0 B/s rd, 127 kB/s wr, 256 op/s rd, 63 op/s wr; 5/2608637 objects degraded (0.000%); 1765938/2608637 objects misplaced (67.696%); 62992 kB/s, 15 objects/s recovering
```

- 若是在 ceph 日志中发现打印如下异常日志，原因是集群非 handy 以及非主 moniter 节点的存储网络断开；

```
2017-05-12 16:05:14.585496 mon.0 172.31.1.31:6789/0 106035 : cluster [INF] osd.31 marked itself down
2017-05-12 16:05:15.095824 mon.0 172.31.1.31:6789/0 106038 : cluster [INF] osd.33 marked itself down
2017-05-12 16:05:15.195542 mon.0 172.31.1.31:6789/0 106040 : cluster [INF] osdmap e286: 36 osds: 25 up, 36 in
2017-05-12 16:05:15.287350 mon.0 172.31.1.31:6789/0 106042 : cluster [INF] osd.27 marked itself down
2017-05-12 16:05:16.186527 mon.0 172.31.1.31:6789/0 106043 : cluster [INF] osdmap e287: 36 osds: 24 up, 36 in
```

/var/log/ceph/ceph-osd.*.log

ceph-osd.*.log 主要记录集群对应硬盘的信息，若集群硬盘出现问题，对应的 OSD 日志将会记录异常原因，作为定位问题的依据。

- OSD 异常（界面显示硬盘异常）时依据 ceph-osd.*.log 的定位过程举例：
- 后台使用命令 ceph osd tree，查看异常硬盘的硬盘标识符；
- 进入相应的硬盘的日志（/var/log/ceph/ceph-osd.*.log）中查看硬盘异常的原因：
- 若是在 ceph-osd 日志中发现打印如下异常日志，原因是 RAID 卡损坏导致 journal 中断：

```
2017-04-25 14:34:08.807146 7f5bf690a780 -1 journal Unable to read past sequence 301115833 but header indicates the journal has committed up through 301115842, journal is corrupt
```

- 若是在 ceph-osd 日志中发现打印如下异常日志，原因是 OSD 压力过大而自杀：

```
2017-03-09 11:46:01.576034 7f0878364700 1 heartbeat_map is_healthy 'FileStore::op_tp thread 0x7f086fa6c700' had suicide timed out after 180
2017-03-09 11:46:01.576049 common/HeartbeatMap.cc: 81: FAILED assert(0 == "hit suicide timeout")
```

- 若是在 ceph-osd 日志中发现打印如下异常日志，原因是 OSD 没有 mount：

```
2017-04-27 19:46:18.280510 7fcfb954c700 5 filestore(/var/lib/ceph/osd/ceph-85) umount /var/lib/ceph/osd/ceph-85
```

- 若是在 ceph-osd 日志中发现打印如下异常日志，原因是数据副本间不一致;

```
2016-10-22 06:49:23.854201 7fd2e860f700- 1 log_channel(cluster)log [ERR]:1.ad shard 1:soid 819850ad/rbd_date.3b7055757a07.0000000000000ab1/7//1 date_digest 0xd7ac1812 != best guess date_digest 0x43d61c5d from auth shard 0
2016-10-22 06:49:23.854253 osd/osd_types.cc:4148:FAILED assert(clone_size.count(clone))
```

/var/log/ceph/ceph-disk.log

ceph-disk.log 主要记录部署 OSD 以及启动 OSD 相关内容，一般与 ceph-osd.*.log 配合来定位 OSD 相关异常问题；

- 若是在 ceph-disk 日志中发现打印如下异常日志，原因是 OSD 激活挂载时，挂载目录“/var/lib/ceph/osd/ceph-*”下存在文件，osd 停止挂载进程退出；问题出现的时间点一般在主机重启时，所有的 OSD 需要重新激活，在 OSD mount 前会检查 OSD 目录下是否有除 heartbeat，osd_disk_info.ini 和 osd_should_be_restart_flag 文件以外的文件，若有其他文件，OSD 停止 mount；

```
ceph-disk: Error: another ceph osd.71 already mounted in position(old/different cluster instance?);unmounting ours.
```

- 若是在 ceph-disk 日志中发现打印如下异常日志，原因是 osd 未激活，未进行挂载；

```
Fri. 07 Apr 2017 10:24:48 ceph-disk[line:2438] ERROR Failed to activate
Fri. 07 Apr 2017 10:24:48 ceph-disk[line:976] DEBUG Unmounting /var/lib/ceph/tmp/mnt.hD_6nh
```

/var/log/ceph/ceph-mon.*.log

ceph-mon.*.log 主要记录集群对应监控节点的信息，monitor 的作用主要是监控集群；若集群监控节点出现问题，对应的 mon 日志将会记录异常原因，作为定位问题的依据。

- mon 异常（界面显示监控节点异常）查看方法：
- 在主机管理中查看异常监控节点的主机名；
- 在后台进入相应的主机的 ceph-mon 日志中查看 mon 异常的原因，ceph-mon 日志对应的查看路径：/var/log/ceph/ceph-mon.*.log。若是在 ceph-mon.*.log 日志中发现打印如下异常日志，原因是主 mon 节点异常（常见原因是主 mon 节点业务网异常或主 mon 节点 ceph-mon 进程停止）,备 mon 触发选举机制；

```
2017-05-08 19:24:58.017935 7fb173765700 1 mon.cvknode84@2(peon).paxos(paxos active c 24348..24883) lease_timeout -- calling new election
2017-05-08 19:24:58.024456 7fb172f64700 0 log_channel(cluster) log [INF] : mon.cvknode84 calling new monitor election
```

/var/log/calamari/calamari.log

calamari.log 日志主要记录的是 handy 界面的操作日志。在 handy 界面对集群进行操作，后台的 calamari.log 日志内容会有相应的记录。

若是在 calamari.log 日志中发现打印如下异常日志，原因是 handy 节点与其他节点网络不通；

```
2017-05-08 15:08:29,060 - ERROR - onestor_common.py[network_check][line:494] - django.request <network_check> Host "172.16.105.84" is unreachable, retry again...
2017-05-08 15:08:29,060 - ERROR - onestor_common.py[execute][line:622] - django.request [ONEStor] onestor_request_all_node cvknode84:Host is unreachable
```

/var/log/onestor_cli/ onestor_cli.log

若收集实时日志的过程中，节点报错，可以查看 onestor_cli.log 日志，onestor_cli.log 日志记录了收集过程中报错的原因，便于开发定位问题。

- 若是在 onestor_cli.log 日志中发现打印如下异常日志，原因是节点日志收集超过 5G 上限；

```
[2017-05-10 10:47:01,980][WARNING][monitor.py][line:157] We detect the current collecting log size is up to 5GB, ending collecting automatically!
```

- 若是发现节点 onestor_cli.log 日志消失，可能原因是节点日志盘空间满了；

5 分布式存储维护

5.1 集群异常问题的恢复处理

5.1.1 硬盘数据分布不均匀的恢复

1. ONEStor 中数据的分布遵循 crush 算法，但会随机出现 OSD 不均匀的情况。

输入 ceph osd df 查看各个硬盘（OSD）上的数据使用率，如下：

```
ID WEIGHT  REWEIGHT SIZE   USE    AVAIL  %USE VAR
 0 3.62999  1.00000  3714G 43171M  3672G 1.14 1.01
 1 3.62999  1.00000  3714G 40320M  3674G 1.06 0.94
 2 3.62999  1.00000  3714G 40539M  3674G 1.07 0.95
 3 3.62999  1.00000  3714G 48686M  3666G 1.28 1.14
 4 3.62999  1.00000  3714G 42906M  3672G 1.13 1.00
 5 3.62999  1.00000  3714G 41701M  3673G 1.10 0.97
 6 3.62999  1.00000  3714G 48277M  3667G 1.27 1.13
 7 3.62999  1.00000  3714G 40986M  3674G 1.08 0.96
 8 3.62999  1.00000  3714G 36113M  3678G 0.95 0.84
 9 3.62999  1.00000  3714G 38921M  3676G 1.02 0.91
10 3.62999  1.00000  3714G 45670M  3669G 1.20 1.07
11 3.62999  1.00000  3714G 42203M  3672G 1.11 0.99
12 3.62999  1.00000  3714G 43916M  3671G 1.15 1.03
13 3.62999  1.00000  3714G 45223M  3670G 1.19 1.06
14 3.62999  1.00000  3714G 45078M  3670G 1.19 1.05
15 3.62999  1.00000  3714G 45118M  3670G 1.19 1.05
16 3.62999  1.00000  3714G 40637M  3674G 1.07 0.95
17 3.62999  1.00000  3714G 38211M  3676G 1.00 0.89
18 3.62999  1.00000  3714G 46927M  3668G 1.23 1.10
19 3.62999  1.00000  3714G 44923M  3670G 1.18 1.05
20 3.62999  1.00000  3714G 38412M  3676G 1.01 0.90
21 3.62999  1.00000  3714G 42534M  3672G 1.12 0.99
22 3.62999  1.00000  3714G 47997M  3667G 1.26 1.12
23 3.62999  1.00000  3714G 39138M  3675G 1.03 0.91
              TOTAL 89140G  1003G 88136G 1.13
MIN/MAX VAR: 0.84/1.14  STDDEV: 0.09
```

图中%USE 为每个硬盘已使用空间的百分比

若有个别硬盘先快写满而其余大部分硬盘还有较大空间，输入 ceph osd reweight-by-utilization 命令触发重新平衡。

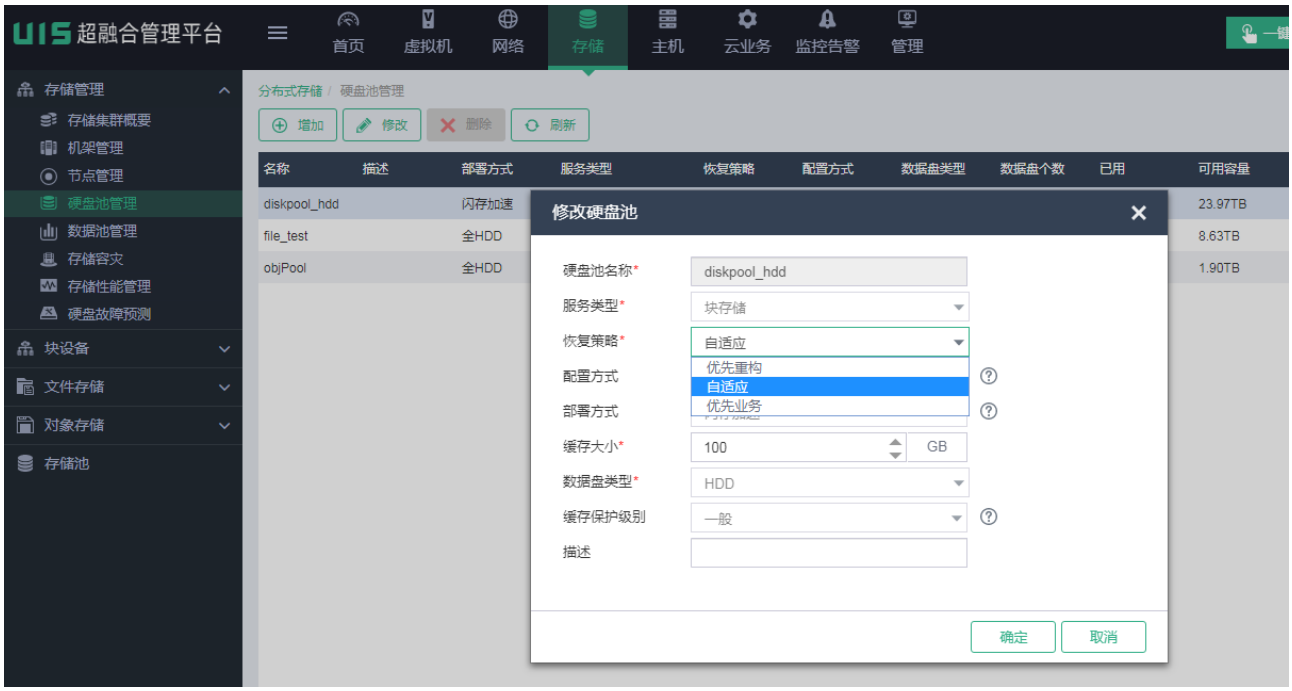
重新平衡的过程中集群的读写压力较大，请注意在非业务高峰期执行。

等待 ceph -s 显示 health HEALTH_OK 后硬盘的数据平衡完成。

2. 集群无业务状态下，可以适当的提高集群数据平衡的速度，方法如下：

- UIS前台调整，操作如下

首先选中需要当前发生了数据均衡的硬盘池，再点击“修改”,出现如下对话框：



“恢复策略”由“自适应”改成“优先重构”。

5.2 节点异常问题的恢复处理

5.2.1 系统盘占满导致的主机异常

系统盘空间可以通过 df -h 查看，若 Use 达到 100%则系统盘被占满，会导致主机异常，比如 apache、ceph 的 mon 进程等无法启动，导致的现象如 mon down，管理节点无法登录等。

```
root@cvknode86:~# df -h
Filesystem                Size  Used Avail Use% Mounted on
/dev/sda1                 28G  4.0G   23G  16% /
```

导致系统盘空间占满的原因及解决方法：

1. 大文件占用、log 日志过多

可以进入/var/log 等相关目录下查看，使用 du -h --max-depth=1 查看当前目录下各个文件夹的大小，删除不需要的文件。

2. fio 测试工具操作失误

执行 fio 时未指定--filename 的情况下，fio 的数据会自动写入系统盘，生成一个 test0.0 的大文件占据大量磁盘空间。echo “”> XXX 然后 rm -rf XXX 删除该文件释放空间即可。

5.3 增删主机或硬盘的过程中网络故障导致的异常

在增删主机或硬盘的过程中该主机出现网络故障，页面检测到之后会弹出以下提示框：

根据网络故障的时间点不同，会有以下三种不同的现象：

5.3.1 硬盘还没有开始删除就出现网络故障

解决方法：等主机网络恢复正常后，再次从页面选择该主机进行删除即可。如果极端情况下主机操作系统损坏不可修复，也可以从页面选择该主机进行离线删除操作，但是主机硬盘上的数据将会残留。

5.3.2 删除掉部分硬盘时出现网络故障

同 5.3.1 的解决方法。

5.3.3 硬盘全部从集群中移除了，但是在格式化硬盘数据的时候出现网络故障

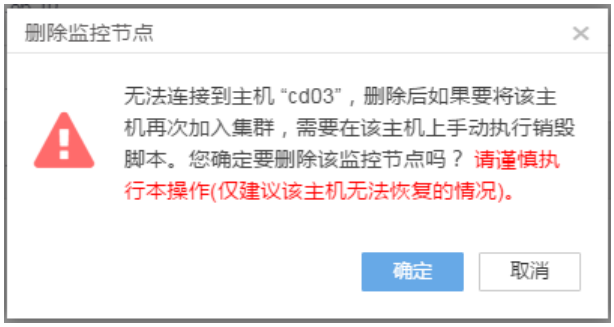
问题现象：在页面上该主机已经不可见了，但是硬盘中的数据和 Ceph 分区会残留，重启主机后这部分硬盘会自动挂载到操作系统上，导致再次增加该主机时这些硬盘无法被扫描到。

解决方法：增加主机之前先手动 umount 掉这些残留的硬盘即可。

```
root@onestor100:~# df
Filesystem     1K-blocks    Used Available Use% Mounted on
/dev/sda1       47346608 3513832  41404668   8% /
none                   4       0         4   0% /sys/fs/cgroup
udev             2012856      12   2012844   1% /dev
tmpfs             404720   46576    358144  12% /run
none                5120       0      5120   0% /run/lock
none             2023600      12   2023588   1% /run/shm
none              102400       0    102400   0% /run/user
/dev/sdb1       94324720   48832  94275888   1% /var/lib/ceph/osd/ceph-2
/dev/sdc1       94324720   45012  94279708   1% /var/lib/ceph/osd/ceph-5
root@onestor100:~#
root@onestor100:~#
root@onestor100:~# umount /var/lib/ceph/osd/ceph-5
```

5.3.4 监控节点离线删除和恢复

监控节点离线删除，是在主机网络无法恢复情况下，将主机从集群彻底删除进行的界面操作。监控节点离线删除是直接从集群删除。



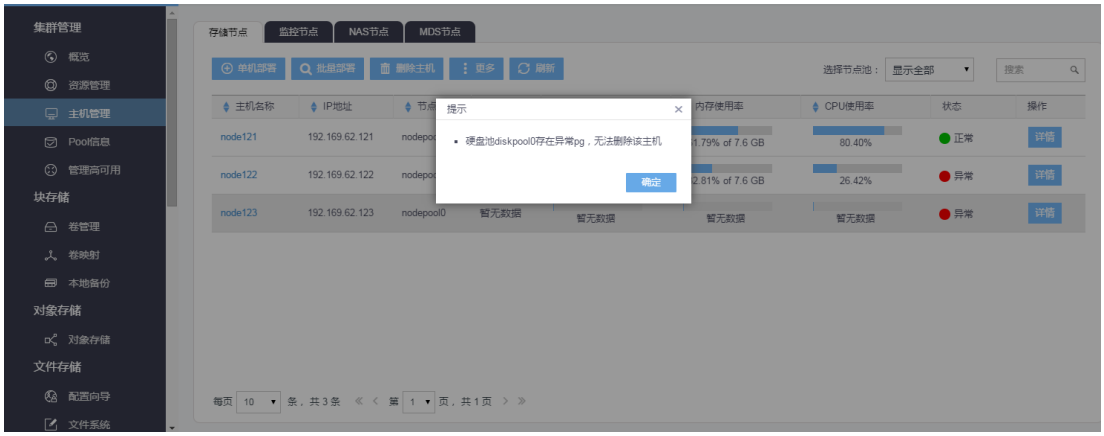
用户若不想让离线的监控节点对集群造成影响，则需要将该主机所有在集群的角色删除，然后销毁该主机，之后可重新加入作为存储、监控或管理高可用节点。

注意：节点销毁操作会造成该节点数据全部破坏，请确认清楚该节点是否已经不再使用！

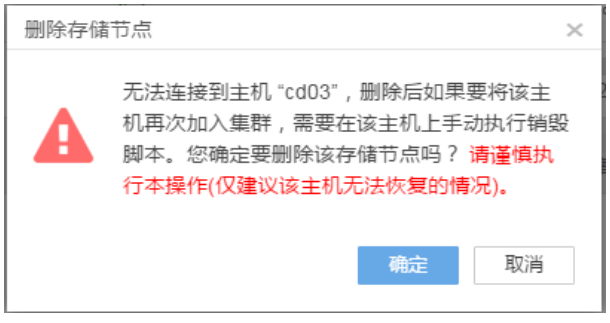
5.3.5 存储节点离线删除和恢复

存储节点离线删除，是在主机网络无法恢复情况下，将主机从集群彻底删除进行的界面操作。存储节点离线删除是直接从集群删除。

在当前节点所属硬盘池存在异常 PG 的情况下，此时可能正在进行数据平衡，为防止数据丢失，请勿此时删除该节点。



在存储节点所属硬盘池健康状态下，则可以正常删除该节点。



用户若不想让离线的存储节点对集群造成影响，则需要将该主机所有在集群的角色删除，然后销毁该主机，之后可重新加入作为存储、监控或管理高可用节点。

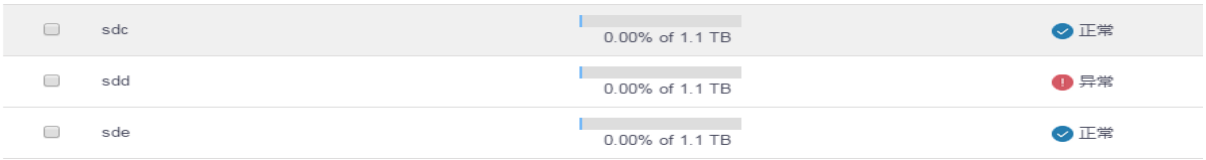
注意：节点销毁操作会造成节点数据全部破坏，请确认清楚该节点是否已经不再使用！

5.4 硬盘异常处理

5.4.1 主机重启导致系统下 sdX 盘号丢失或错位的恢复方法

在拔除硬盘时，RAID 卡上的逻辑分区会由 OK 变为 FAIL，正常操作下 sdX 的盘号不会变化，再使用恢复步骤将逻辑分区从 FAIL 修复为 OK，硬盘即可正常使用。但是，当在逻辑分区 FAIL 时不慎将主机重启，将会造成该硬盘在操作系统上不可见，lsblk 或 fdisk 观察少了一个硬盘。

例如，lsblk 查看原本硬盘为 sda、sdb、sdc、sdd、sde，ONEStor 界面观察 sdd 异常，输入 hpssacli controller all show config 发现 sdd 对应的逻辑分区 FAIL，如下图：




```
array C
   logicaldrive 3 (1.1 TB, RAID 0, OK)
array D
   logicaldrive 4 (1.1 TB, RAID 0, Failed)
```

此时主机意外重启后，sdd 硬盘系统将不可见，后面的 sde 硬盘号向前漂移变为 sdd，使得只能查看到 sda、sdb、sdc、sdd，少了一个硬盘。此时即使将该逻辑分区修复为 OK，丢失的硬盘也不可见。

解决该问题的方法：

(1) 将原本 FAIL 的逻辑分区，不管现在其在 FAIL 或 OK 状态，将其删除

```
hpssacli ctrl slot=0 logicaldrive 4 delete forced
```

(2) 输入 hpssacli controller all show config，找到最后显示 unassigned，未被分配的物理硬盘，如下：

```
unassigned
   physicaldrive 2I:2:3 (port 2I:box 2:bay 3, SAS, 1200.2 GB, OK)
```

(3) 重新创建逻辑分区

```
hpssacli ctrl slot=0 create type=ld drives=2I:2:3 raid=0
```

(4) 此时 lsblk 下查看，该新增盘添加在现有盘号的末尾，为 sde。此时将原本的 OSD 目录重新挂载至/dev/sde1 硬盘分区。

```
mount /dev/sde1 /var/lib/ceph/osd/ceph-4
```

若此时 ONEStor 界面还显示 sde 异常，将 sde 删除后重新添加，即可恢复正常。

5.4.2 查询 OSD 目录所 mount 的数据分区、journal（写加速）分区

正常 mount 状态：

```
sdb                 8:16   0   3.7T  0 disk
├─sdb1              8:17   0   3.6T  0 part /var/lib/ceph/osd/ceph-2
└─sdb2              8:18   0    10G  0 part
sdc                 8:32   0   3.7T  0 disk
├─sdc1              8:33   0   3.6T  0 part /var/lib/ceph/osd/ceph-5
└─sdc2              8:34   0    10G  0 part
sdd                 8:48   0   3.7T  0 disk
├─sdd1              8:49   0   3.6T  0 part /var/lib/ceph/osd/ceph-8
└─sdd2              8:50   0    10G  0 part
sde                 8:64   0   3.7T  0 disk
├─sde1              8:65   0   3.6T  0 part /var/lib/ceph/osd/ceph-11
└─sde2              8:66   0    10G  0 part
```

umount 状态：

```
sdb                 8:16   0   3.7T  0 disk
├─sdb1              8:17   0   3.6T  0 part
└─sdb2              8:18   0    10G  0 part
sdc                 8:32   0   3.7T  0 disk
├─sdc1              8:33   0   3.6T  0 part
└─sdc2              8:34   0    10G  0 part
sdd                 8:48   0   3.7T  0 disk
├─sdd1              8:49   0   3.6T  0 part
└─sdd2              8:50   0    10G  0 part
sde                 8:64   0   3.7T  0 disk
├─sde1              8:65   0   3.6T  0 part
└─sde2              8:66   0    10G  0 part
```

当硬盘异常后想要从 umount 状态恢复至 mount 状态，或者想找到一个 OSD 的 journal（写加速）盘分区，需要通过查询 partuuid 来准确地查询到对应关系。

(1) 查看 OSD 目录下的 fdis 文件，里面记录了 OSD 数据分区的 partuuid

```
cat /var/lib/ceph/osd/ceph-8/fsid
d6d97f59-171e-46f7-9759-8037c7209bf1
```

(2) 查看 OSD 目录下的 journal_uuid 文件，里面记录了 OSD 所对应的 journal 分区的 partuuid

```
cat /var/lib/ceph/osd/ceph-8/journal_uuid
1f8b0b99-69c6-404a-acfe-186f435fd877
```

(3) 查询主机下所有分区的 partuuid

```
ll /dev/disk/by-partuuid/    (下面列出的结果是写缓存 SSD sdf 的值)
lrwxrwxrwx 1 root root  10 Dec  6 19:55 1f8b0b99-69c6-404a-acfe-186f435fd877 -> ../../sdf1
lrwxrwxrwx 1 root root  10 Dec  6 19:55 260c435a-2c35-4562-979d-7a3d641dda48 -> ../../sdf2
```

(4) 找到相同的 partuuid 对应即可

5.4.3 UIS 界面未删除故障 osd，直接更换新盘导致原 osd 无法删除的解决方法

UIS 界面上未删除坏盘的 OSD，直接更换新的硬盘后，Handy 上添加新的硬盘做 OSD，导致原来的 OSD 显示暂无数据，无法删除，此时可以通过后台命令删除该 OSD。

(1) lsblk 查看旧 osd 是否仍然挂载，保证已取消挂载

正常 mount 状态：

```
sdb                 8:16   0   3.7T  0 disk
├─sdb1              8:17   0   3.6T  0 part /var/lib/ceph/osd/ceph-2
└─sdb2              8:18   0    10G  0 part
sdc                 8:32   0   3.7T  0 disk
├─sdc1              8:33   0   3.6T  0 part /var/lib/ceph/osd/ceph-5
└─sdc2              8:34   0    10G  0 part
sdd                 8:48   0   3.7T  0 disk
├─sdd1              8:49   0   3.6T  0 part /var/lib/ceph/osd/ceph-8
└─sdd2              8:50   0    10G  0 part
sde                 8:64   0   3.7T  0 disk
├─sde1              8:65   0   3.6T  0 part /var/lib/ceph/osd/ceph-11
└─sde2              8:66   0    10G  0 part
```

umount 状态：

```
sdb                 8:16   0   3.7T  0 disk
├─sdb1              8:17   0   3.6T  0 part
└─sdb2              8:18   0    10G  0 part
sdc                 8:32   0   3.7T  0 disk
├─sdc1              8:33   0   3.6T  0 part
└─sdc2              8:34   0    10G  0 part
sdd                 8:48   0   3.7T  0 disk
├─sdd1              8:49   0   3.6T  0 part
└─sdd2              8:50   0    10G  0 part
sde                 8:64   0   3.7T  0 disk
├─sde1              8:65   0   3.6T  0 part
└─sde2              8:66   0    10G  0 part
```

(2) 通过 ps –ef |grep osd 查看旧 osd 进程是否停止

(3) 通过后台指令停止 osd 进程，x 为 osd 进程编号

```
stop ceph-osd id=x
ceph osd out osd.x
ceph osd crush remove osd.x
ceph auth del osd.x
ceph osd rm osd.x
```

需要注意的是，此类命令会直接清除用户数据，请谨慎使用，如果有疑问联系总部。

(4) 通过 cephosd tree 查看 OSD 是否成功从集群移除

(5) 登录 UIS 界面查看故障磁盘已删除。

5.5 硬盘更换

参见《UIS 超融合一体机部件更换配置指导》中的硬盘部分。

6 典型问题排查与处理

6.1 UIS标准版集群初始化失败问题

6.1.1 无法扫描到主机

1. 网络排查

a) 确保主机管理接口与 Manager 节点的管理口处于同一局域网。

b) 主机管理接口对应的交换机端口配置了端口聚合

如果配置了静态端口聚合，需要 shutdown 其中一个端口，待主机扫描完成后再 up 端口；

如果配置了动态端口聚合，需要配置端口为边缘端口（lacp edge-port）。

2. 集群曾经初始化失败过

需要查看每个 cvk 中是否有文件残留。查看 /etc/cvk 下的 cvm_info 文件和/root/.ssh 下的 mhost 文件，如果文件存在，需要手动删除。

```
rm -rf cvm_info
rm -rf mhost
```

3. 要加入的主机曾经做过 manager 节点

查看/root/.ssh 下是否存在 isCvmFlag 文件，如果文件存在，需要手动删除。

```
rm -rf isCvmFlag
```

6.1.2 创建集群失败

创建计算集群失败，一般为网络异常，需排查现场网络环境。确保所有主机的管理网、存储外网和存储内网都可达，可以配置上 ip 后检测连通性。

6.1.3 配置存储失败

1. 扫描到的磁盘不全或者扫描不到指定的磁盘

(1) 磁盘有分区造成

在主机后台使用命令 lsblk 查看磁盘是否存在分区，如果存在需要删除对应分区 parted /dev/sdx rm y （x 为盘符，y 为分区号）。

举例：删除 sdd 磁盘的第三个分区：parted /dev/sdd rm 3

(2) Raid 卡规格不支持

RAID 卡型号需要在《H3C CAS&UIS 服务器虚拟化产品软硬件兼容性列表》里。

2. 管理节点没有正确安装分布式存储

这种情况可以通过后台重新安装 onestor 的方式进行尝试解决，具体是执行脚本 /opt/bin/uis_onestor_handy_install.sh 脚本

如果执行该脚本后然后报错，请联系技术支持人员。

3. 服务器或者 raid 卡不支持设备管理

- UIS 0716 版本以前版本：修改 handy 节点的 check_raid_support 这个脚本,，屏蔽设备管理，执行命令 sed –i ‘s/\$result/false/g’ /opt/h3c/sbin/check_raid_support，然后执行 check_raid_support 返回 false，即可。
- UIS0716 之后版本：修改 handy 节点的/opt/h3c/sbin/devmgr_check_dev_type 这个脚本, 屏蔽设备管理，在 def check_raid_card()函数中，直接添加一行代码:return False。

```
import os
import platform
import re
import base64
from M2Crypto import RSA

def check_raid_card():
    """
    DM_ONEstor_list 中目前只支持LSI的raid卡
    return: True or False
    """
    return False
    ret = False
    count = 0
    try:
        # Intel Corporation Device 201d是虚拟的，排除掉;有部分RAID卡显示为Serial Attached SCSI controller
        raid_cmd = "lspci 2>/dev/null| grep -E 'SCSI controller|RAID bus controller'|grep -v 'Intel Corporation Device 201d'"
        cmd_ret = os.popen(raid_cmd).read().strip().split('\n')
        num = len(cmd_ret)
        # 暂不限制多张RAID卡的场景
        if num > 1 or num < -1:
            return ret
        nvme_cmd = "lspci 2>/dev/null| grep 'Non-Volatile' | wc -l"
        nvme_ret = os.popen(nvme_cmd).read()
        if nvme_ret and 6 < int(nvme_ret):
            return ret
        for raid_info in cmd_ret:
            for support_type in support_raid_list:
                if raid_info.find(support_type) > -1:
                    count = count + 1

        if count == num:
            ret = True
    except:
        ret = False

    return ret

def check_specific_local():
```

然后执行 devmgr_check_dev_type 返回如下信息（for_DM_ONEstor 为 False），即可。

```
[root@cvknode1 ~]# devmgr_check_dev_type
cat: /etc/.onekey: No such file or directory
{'for_install': False, 'x10000_type': 'UIS-Cell 3020 G3', 'for_DM_ONEstor': False, 'is_X10000': False}
```

6.2 集群状态相关

6.2.1 健康度不到 100%

1. 节点故障或网络不通

(1) 登录 UIS 前台界面查看告警以及主机状态是否存在故障；

(2) 登录 UIS 后台，使用 ping 方式查看集群中主机间的连通性。

2. 硬盘故障或 RAID 卡故障

(1) 登录 UIS 前台界面查看告警是否存在硬盘或者 RAID 卡故障；

(2) 登录 HDM 查看是否存在硬件告警。

6.3 删除主机相关

6.3.1 删除主机提示删除失败，实际删除成功

1. 问题定位

(1) 在该主机上执行 lsblk 查看是否有 osd 未被 umount

```
root@unistor2:/var/lib/ceph/osd/ceph-11# lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0   100G  0 disk
├─sda1   8:1    0  60.9G  0 part /
├─sda2   8:2    0     1K  0 part
├─sda5   8:5    0  35.1G  0 part /var/log
└─sda6   8:6    0     4G  0 part [SWAP]
sdb      8:16   0    16G  0 disk
sdc      8:32   0    16G  0 disk
sdd      8:48   0    16G  0 disk
sde      8:64   0    16G  0 disk
sdf      8:80   0    16G  0 disk
└─sdf1   8:81   0   100M  0 part /var/lib/ceph/osd/ceph-11
sdg      8:96   0    16G  0 disk
sdh      8:112  0    16G  0 disk
sr0     11:0    1   1.5G  0 rom
```

(2) 可以看到有分区残留，查看该 osd 目录是否被打开，可以确定该问题是由于删除主机时打开了主机下的 osd 目录。

```
/var/lib/ceph/osd/ceph-11
```

2. 解决方法

使用 cd 命令退出该 osd 目录，然后手动执行 umount /var/lib/ceph/osd/ceph-11 即可

之后执行 sgdisk –zap-all /dev/sdf 格式化分区。

```
root@unistor2:~# sgdisk --zap-all /dev/sdf
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
root@unistor2:~# lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0   100G  0 disk
├─sda1   8:1    0  60.9G  0 part /
├─sda2   8:2    0     1K  0 part
├─sda5   8:5    0  35.1G  0 part /var/log
└─sda6   8:6    0     4G  0 part [SWAP]
sdb      8:16   0    16G  0 disk
sdc      8:32   0    16G  0 disk
sdd      8:48   0    16G  0 disk
sde      8:64   0    16G  0 disk
sdf      8:80   0    16G  0 disk
sdg      8:96   0    16G  0 disk
sdh      8:112  0    16G  0 disk
sr0     11:0    1   1.5G  0 rom
```

6.4 硬盘扩容问题

6.4.1 无可用的硬盘

1. 问题定位

查看节点内 osd 是否已被 ceph 集群使用过。使用 lsblk 查看想要添加的硬盘，查看硬盘已有分区，再使用 gdisk -l /dev/xxx (xxx 为盘符名称)命令，查看硬盘分区中有 ceph 标识，则认为此硬盘已被使用。

```
root@unode76:~# gdisk -l /dev/sdc
GPT fdisk (gdisk) version 0.8.8

Partition table scan:
  MBR: protective
  BSD: not present
  APM: not present
  GPT: present

Found valid GPT with protective MBR; using GPT.
Disk /dev/sdc: 209715200 sectors, 100.0 GiB
Logical sector size: 512 bytes
Disk identifier (GUID): 366665E4-F435-4A42-947D-99E0CC96E20E
Partition table holds up to 128 entries
First usable sector is 34, last usable sector is 209715166
Partitions will be aligned on 2048-sector boundaries
Total free space is 2014 sectors (1007.0 KiB)

Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048          206847   100.0 MiB   FFFF  ceph data
   2          206848       209715166   99.9 GiB    FFFF  ceph block
root@unode76:~#
```

2. 处理方法

(1) 若确认此硬盘未被用户使用，只是之前残留导致，则使用 ceph-disk zap /dev/xxx(xxx 为盘符名称)清除残留数据 后，再尝试添加。

```
sde      8:64   0   100G  0 disk
├─sde1   8:65   0   100M  0 part
└─sde2   8:66   0  99.9G  0 part
sr0     11:0    1  1024M  0 rom
root@unode78:~# ceph-disk zap /dev/sde
1+0 records in
1+0 records out
512 bytes (512 B) copied, 0.0162661 s, 31.5 kB/s
1+0 records in
1+0 records out
512 bytes (512 B) copied, 0.00118326 s, 433 kB/s
Caution: invalid backup GPT header, but valid main header; regenerating
backup header from main header.

Warning! Main and backup partition tables differ! Use the 'c' and 'e' options
on the recovery & transformation menu to examine the two tables.

Warning! One or more CRCs don't match. You should repair the disk!

********************************************************************************
Caution: Found protective or hybrid MBR and corrupt GPT. Using GPT, but disk
verification and recovery are STRONGLY recommended.
********************************************************************************
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
Creating new GPT entries.
The operation has completed successfully.
root@unode78:~# lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0   100G  0 disk
├─sda1   8:1    0  59.6G  0 part /
├─sda2   8:2    0     1K  0 part
├─sda5   8:5    0  34.4G  0 part /var/log
└─sda6   8:6    0     6G  0 part [SWAP]
sdb      8:16   0   100G  0 disk
├─sdb1   8:17   0   100M  0 part /var/lib/ceph/osd/ceph-6
└─sdb2   8:18   0  99.9G  0 part
sdc      8:32   0   100G  0 disk
├─sdc1   8:33   0   100M  0 part /var/lib/ceph/osd/ceph-9
└─sdc2   8:34   0  99.9G  0 part
sdd      8:48   0   100G  0 disk
├─sdd1   8:49   0   100M  0 part /var/lib/ceph/osd/ceph-1
└─sdd2   8:50   0  99.9G  0 part
sde      8:64   0   100G  0 disk
sr0     11:0    1  1024M  0 rom
```

当前 UIS 最新版本已经支持前台清理分区，如果清理分区后仍然无法扫描到磁盘，可以尝试再次执行 ceph-disk zap /dev/xxx

(2) 如果主机中存在不支持设备管理的主机，需要手动关闭设备管理，同时需要集群中节点对设备管理的支持保持一致，例如 handy 节点不支持设备管理，需要新扩容的主机也不支持设备管理。

(3) 设备管理处理方法：

- UIS 0716 版本以前版本：修改 handy 节点的 check_raid_support 这个脚本,，屏蔽设备管理，执行命令 sed –i ‘s/\$result/false/g’ /opt/h3c/sbin/check_raid_support，然后执行 check_raid_support 返回 false，即可。
- UIS0716 之后版本：修改 handy 节点的/opt/h3c/sbin/devmgr_check_dev_type 这个脚本, 屏蔽设备管理，在 def check_raid_card()函数中，直接添加一行代码:return False。

```
import os
import platform
import re
import base64
from M2Crypto import RSA

def check_raid_card():
    """
    DM_ONEstor_list 中目前只支持LSI的raid卡
    return: True or False
    """
    return False
    ret = False
    count = 0
    try:
        # Intel Corporation Device 201d是虚拟的，排除掉;有部分RAID卡显示为Serial Attached SCSI controller
        raid_cmd = "lspci 2>/dev/null| grep -E 'SCSI controller|RAID bus controller'|grep -v 'Intel Corporation Device 201d'"
        cmd_ret = os.popen(raid_cmd).read().strip().split('\n')
        num = len(cmd_ret)
        # 暂不限制多张RAID卡的场景
        if num > 1 or num < -1:
            return ret
        nvme_cmd = "lspci 2>/dev/null| grep 'Non-Volatile' | wc -l"
        nvme_ret = os.popen(nvme_cmd).read()
        if nvme_ret and 6 < int(nvme_ret):
            return ret
        for raid_info in cmd_ret:
            for support_type in support_raid_list:
                if raid_info.find(support_type) > -1:
                    count = count + 1

        if count == num:
            ret = True
    except:
        ret = False

    return ret

def check_specific_local():
```

然后执行 devmgr_check_dev_type 返回如下信息（for_DM_ONEstor 为 False），即可。

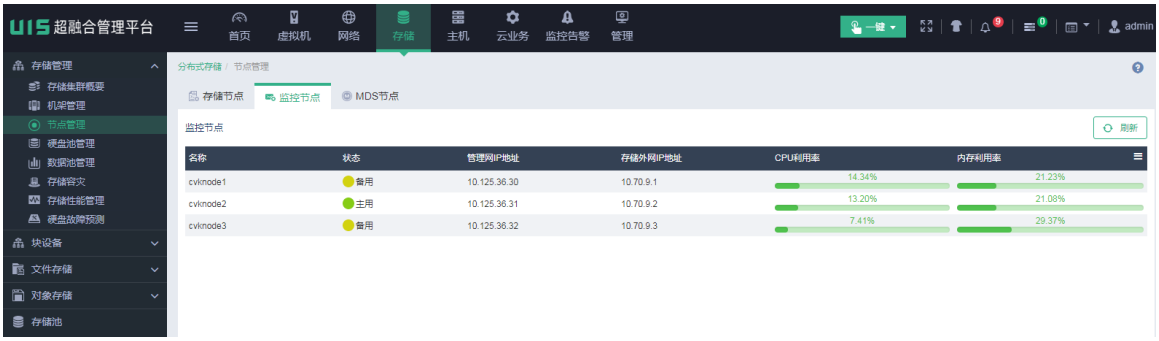
```
[root@cvknode1 ~]# devmgr_check_dev_type
cat: /etc/.onekey: No such file or directory
{'for_install': False, 'x10000_type': 'UIS-Cell 3020 G3', 'for_DM_ONEstor': False, 'is_X10000': False}
```

6.5 集群常见告警及处理

1. mon 节点 down“1 mons down”

原因：mon 节点所在主机掉电、关机、网络异常。

在”存储” ->”节点管理”->“监控节点”，检查监控节点的状态：



检查异常的监控节点是否掉电、关机，然后检查暂无数据的主机与集群之间网络是否正常。

2. osd 状态为 down，例如“3 osds are down”

原因：

（1） osd 所在的主机掉电、关机、网络异常。

在”存储”->”节点管理”->“存储节点”，检查存储节点状态，如果主机掉电、关机、业务网络异常存储节点会显示暂无数据，如下图：

检查暂无数据的主机是否掉电、关机以及主机与集群之间网络是否正常。

（2） osd 进程异常关闭

在”存储”-”节点管理”->“存储节点”，检查存储节点硬盘状态是否正常 .

使用管理网 ssh 登录存储节点异常的主机 IP，输入命令行“ceph osd tree”显示所有的 osd 状态：

```
root@upgrade04:~# ceph osd tree
ID WEIGHT  TYPE NAME             UP/DOWN REWEIGHT PRIMARY-AFFINITY
-6       0 root ssd_root
-5       0     rack rack0_ssd
-1 0.11993 root default
-4 0.11993     rack rack0
-2 0.02998         host upgrade01
 1 0.00999             osd.1           up  1.00000          1.00000
 4 0.00999             osd.4           up  1.00000          1.00000
 7 0.00999             osd.7           up  1.00000          1.00000
-3 0.02998         host upgrade02
 2 0.00999             osd.2           up  1.00000          1.00000
 5 0.00999             osd.5           up  1.00000          1.00000
 8 0.00999             osd.8           up  1.00000          1.00000
-7 0.02998         host upgrade03
 3 0.00999             osd.3           up  1.00000          1.00000
 6 0.00999             osd.6           up  1.00000          1.00000
 0 0.00999             osd.0           up  1.00000          1.00000
-8 0.02998         host upgrade04
 9 0.00999             osd.9           up  1.00000          1.00000
10 0.00999             osd.10        down        0          1.00000
11 0.00999             osd.11        down        0          1.00000
root@upgrade04:~#
```

查看所有 osd 进程是否已启动“ps-ef |grep ceph-osd”：

```
root@upgrade04:~# ps -ef | grep ceph-osd
root      1014     1  0 10:53 ?        00:00:08 /usr/bin/ceph-osd --cluster=ceph -i 9 -f
root     16973  3992  0 11:19 pts/3    00:00:00 grep --color=auto ceph-osd
root@upgrade04:~#
```

将未启动的 osd 进程手动启动“systemctl start ceph-osd@xx.service（xx 为 osd 的 id 编号）”：

```
root@upgrade04:~# start ceph-osd id=10
ceph-osd (ceph/10) start/running, process 18463
root@upgrade04:~# start ceph-osd id=11
ceph-osd (ceph/11) start/running, process 18753
root@upgrade04:~# ps -ef | grep ceph-osd
root      1014     1  0 10:53 ?        00:00:09 /usr/bin/ceph-osd --cluster=ceph -i 9 -f
root     18463     1 10 11:22 ?        00:00:01 /usr/bin/ceph-osd --cluster=ceph -i 10 -f
root     18753     1 10 11:22 ?        00:00:00 /usr/bin/ceph-osd --cluster=ceph -i 11 -f
root     18965  3992  0 11:22 pts/3    00:00:00 grep --color=auto ceph-osd
root@upgrade04:~# ceph osd tree
ID WEIGHT  TYPE NAME             UP/DOWN REWEIGHT PRIMARY-AFFINITY
-6       0 root ssd_root
-5       0     rack rack0_ssd
-1 0.11993 root default
-4 0.11993     rack rack0
-2 0.02998         host upgrade01
 1 0.00999             osd.1           up  1.00000          1.00000
 4 0.00999             osd.4           up  1.00000          1.00000
 7 0.00999             osd.7           up  1.00000          1.00000
-3 0.02998         host upgrade02
 2 0.00999             osd.2           up  1.00000          1.00000
 5 0.00999             osd.5           up  1.00000          1.00000
 8 0.00999             osd.8           up  1.00000          1.00000
-7 0.02998         host upgrade03
 3 0.00999             osd.3           up  1.00000          1.00000
 6 0.00999             osd.6           up  1.00000          1.00000
 0 0.00999             osd.0           up  1.00000          1.00000
-8 0.02998         host upgrade04
 9 0.00999             osd.9           up  1.00000          1.00000
10 0.00999             osd.10          up  1.00000          1.00000
11 0.00999             osd.11          up  1.00000          1.00000
root@upgrade04:~#
```

（3） OSD软连接丢失

先使用 lsblk 命令找到 down 的硬盘对应的 osd 目录，如下

```
root@cvknode2:~# lsblk
NAME                                    MAJ:MIN RM   SIZE RO TYPE  MOUNTPOINT
sda                                       8:0    0 279.4G  0 disk
├─sda1                                    8:1    0    28G  0 part  /
├─sda2                                    8:2    0     1K  0 part
├─sda5                                    8:5    0  18.6G  0 part  /var/log
├─sda6                                    8:6    0  91.6G  0 part  [SWAP]
└─sda7                                    8:7    0 141.3G  0 part  /vms
sdb                                       8:16   0 279.4G  0 disk
sdc                                       8:32   0   3.7T  0 disk
├─sdc1                                    8:33   0   3.6T  0 part  /var/lib/ceph/osd/ceph-1
└─sdc2                                    8:34   0    10G  0 part
sdd                                       8:48   0   3.7T  0 disk
├─sdd1                                    8:49   0   3.6T  0 part  /var/lib/ceph/osd/ceph-2
└─sdd2                                    8:50   0    10G  0 part
sde                                       8:64   0   3.7T  0 disk
├─sde1                                    8:65   0   3.6T  0 part  /var/lib/ceph/osd/ceph-3
└─sde2                                    8:66   0    10G  0 part
sdf                                       8:80   0   3.7T  0 disk
├─sdf1                                    8:81   0   3.6T  0 part  /var/lib/ceph/osd/ceph-4
└─sdf2                                    8:82   0    10G  0 part
```

进入该目录

cd /var/lib/ceph/osd/ceph-4

输入 ll 查看软连接是否存在，正常如下，journal 文件对应了一个 disk 的 uuid

```
root@cvknode2:/var/lib/ceph/osd/ceph-4# ll
total 52
drwxr-xr-x  3 root root  233 Jun 21 18:06 ./
drwxr-xr-x 10 root root 4096 Jun 21 18:09 ../
-rw-r--r--  1 root root  194 Jun 21 18:06 activate.monmap
-rw-r--r--  1 root root    3 Jun 21 18:06 active
-rw-r--r--  1 root root   37 Jun 21 18:06 ceph_fsid
drwxr-xr-x 71 root root 1167 Jun 21 18:11 current/
-rw-r--r--  1 root root   37 Jun 21 18:06 fsid
-rw-r--r--  1 root root    1 Jun 22 23:52 heartbeat
lrwxrwxrwx  1 root root   58 Jun 21 18:06 journal -> /dev/disk/by-partuuid/f77725ae-3f03-4805-ac71-43e413c062d4
-rw-r--r--  1 root root   37 Jun 21 18:06 journal_uuid
-rw-------  1 root root   56 Jun 21 18:06 keyring
-rw-r--r--  1 root root   21 Jun 21 18:06 magic
-rw-r--r--  1 root root    6 Jun 21 18:06 ready
-rw-r--r--  1 root root    4 Jun 21 18:06 store_version
-rw-r--r--  1 root root   53 Jun 21 18:06 superblock
-rw-r--r--  1 root root    0 Jun 21 18:08 upstart
-rw-r--r--  1 root root    2 Jun 21 18:06 whoami
```

若这个软连接不存在，请输入一下命令进行修复

ceph-disk activate-all

（4） 硬盘松动、故障

如果某个硬盘故障，则对应 OSD 进程 down。此时可以通过观察服务器硬盘故障灯来确认，进行硬盘更换。

3. pg 状态告警，例如“32 pgs degraded”“108 pgs stale”“15 pgs stuck unclean”“32 pgs undersized”

若此时无其他告警，表明数据正在迁移，一段时间后 pg 状态会自动恢复正常。

4. 缓存告警

缓存告警包括物理缓存告警和逻辑缓存告警，一般产生告警的原因主要有两个：第一是在开局的时候如果是手动做的 raid，可能没有按照缓存设置标准去开启关闭缓存，第二是在集群使用过程中的故障造成，例如 raid 卡电池故障可能引起的逻辑缓存异常等。

此时可执行一键巡检，然后进行缓存修复

(1) 执行一键巡检，勾选“物理磁盘状态”和“逻辑磁盘状态”。

(2)　点击对应磁盘的故障按钮，当前页面自动调整到底部的修复部分

(3)　点击“修复”进行缓存的修复。

6.6 UIS Manager主机故障恢复方法

6.6.1 通过 UIS Manager 备份数据恢复

H3C UIS 超融合管理平台支持通过管理员手工对管理节点的配置进行备份，或者制定合适的备份策略，定时备份管理节点的配置数据，保证管理平台的高可用性。当管理节点服务器出现故障无法恢复后，使用另外的服务器重新安装 H3C UIS 超融合管理平台，再导入先前备份的配置数据，恢复虚拟化业务管理功能，确保 H3C UIS 超融合管理平台的故障不会影响到虚拟化环境的管理。

当 H3C UIS 超融合管理平台所在的服务器故障后，需要在备用的服务器上重新安装 H3C UIS 超融合管理平台，此时，先前备份的 UIS Manager 配置将被导入到新的 H3C UIS 超融合管理平台。

如下是 UIS Manager 主机故障时的还原操作步骤：

(1) 在备用的服务器重新安装 H3C UIS 超融合管理平台完成后，系统管理员通过浏览器访问 H3C UIS 超融合管理平台，在导航菜单中依次选择【管理】/【管理数据备份】，在“管理数据备份”标签页下，配置成备份服务器上存储备份文件的位置。

(2) 配置完成后单击连接测试按钮，进行连通性检测，测试成功后点击保存按钮进行保存。（如果测试失败请查看地址、用户密码是否正确，备份路径是否存在。）

(3) 点击“备份历史”标签页，H3C UIS 超融合管理平台将自动从指定的备份位置获取所有的备份历史信息

(4) 在“备份历史”中，选择需要恢复的 UIS Manager 配置数据，点击该数据所在行的“ ”图标，在弹出的确认话框中选择<是>按钮。

(5) 还原完成后清空浏览器缓存，再重新进行登陆。

注意

系统盘本身就自带备份。一块盘损坏不影响系统的正常运行。如果两块盘都损坏，则系统也无法恢复。所以当一个系统盘损坏之后请及时更换。

6.7 双机相关

6.7.1 仲裁主机故障恢复

详见《H3C UIS 超融合产品双机热备配置指导》。

6.8 mon异常修复

6.8.1 系统盘空间利用率过高导致的 mon down

1. 问题定位

(1) 查看 mon 进程是否存在

```
ps -ef|grep ceph-mon
```

(2) 若 mon 进程没有启动，则 df -h，查看系统盘占用率

(3) df -h 查看系统盘占用情况

```
root@cvknode1:df -h
Filesystem Size Used Avail Use% Mounted on
/dev/sda1 10G 9.6G 0.4G 96% /
udev 863M 12K 863M 1% /dev
tmpfs 349M 348K 349M 1% /run
none 5.0M 0 5.0M 0% /run/lock
none 873M 4.0K 873M 1% /run/shm
```

(4) 查看进程状态：ps aux | grep ceph-mon

```
root@cvknode20216:~/515# ps aux | grep ceph-mon
root 2619507 0.0 0.1 8112 2136 pts/3 S+ 17:47 0:00 grep --color=auto ceph-mon
```

系统盘占用超过 95%，mon 进程会退出或者起不来；事实上，系统盘占用大于等于 70%，会提示 low disk space；占用大于等于 95%，mon 进程异常。

2. 解决方法

可通过释放系统盘空间，启动 mon 进程解决，例如 service ceph-mon@cvknode2 status（不同节点服务名称有差异）。

6.8.2 网络错误导致的 mon down

1. 问题定位

(1) 查看 mon 进程是否启动

(2) 若 mon 进程存在，则测试 mon 之间的互 ping 是否正常

(3) 通过 arp -a 和 ifconfig 查看 mon 节点的 arp 表是否打印正确，

2. 解决方法

解决网络问题后，启动 mon 进程恢复。

6.9 extent备份恢复文件

6.9.1 检测是否开启 extent 备份

检查下版本是否开启 extent 备份，红色标注部分代表 12 小时备份一次，如未开启 extent 备份，无法使用该方法恢复文件

```
cat /etc/crontab
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
```

```
MAILTO=""

# For details see man 4 crontabs

# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name  command to be executed

0 22    * * 5   root    python /opt/bin/ocfs2_pool_fstrim.pyc -s onestor
1 2     * * *   root    /opt/bin/cas_clean_log.sh
*/1 *   * * *   root    python /opt/bin/uis_host_network_probe.pyc
*/5     *       *       *       *       root    flock   -xn     /tmp/util_memory_dropcaches.sh.lock     -c
"/opt/bin/util_memory_dropcaches.sh"
*/3 * * * *   root    /opt/bin/check_abrt_memory.sh
* *     * * *   root    /opt/bin/ocfs2_iscsi_conf_chg_timer.sh
*/10 *    * * *   root    python /opt/bin/ocfs2_cluster_config.pyc -s
0 */12    * * *   root    python /opt/bin/ocfs2_filesystem_layout_backup.pyc
* *     * * *   root    /opt/bin/tomcat_check.sh
*/10 * * * *   root    /opt/bin/ntp_mon.sh
* *     * * *   root    /opt/bin/tomcat_check.sh
```

6.9.2 extent 备份目录

备份文件在目录/vms/.ocfs2_extent_backup 下，通过文件名查找对应 lzo 文件，defaultPool_hdd 为存储池名，根据时间找最近的一个 extent 备份文件

```
ll –a  /vms/.ocfs2_extent_backup/defaultPool_hdd/normal/
-rw-r--r-- 1 root root 176 Dec 24 00:00 .8257798_root_zhanji_1_202012240000.lzo
比如：
/vms/.ocfs2_extent_backup/defaultPool_hdd/normal/.8257798_root_zhanji_1_202012240000.lzo
```

6.9.3 extent 备份文件解压

extent 备份拷贝到其他目录（比如 home），再解压

```
cp
/vms/.ocfs2_extent_backup/defaultPool_hdd/normal/.8257798_root_zhanji_1_202012240000.lzo
/home
cd /home
lzop -dv .8257798_root_zhanji_1_202012240000.lzo
```

6.9.4 使用脚本恢复文件

```
python /opt/bin/ocfs2_restore_utils.pyc dd /dev/dm-0 /home/.8257798_root_zhanji_1_202012240000
/vms/hw235-1/8257798_root_zhanji_1_202012240000_new
```

注：
/dev/dm-0:被恢复文件所在共享存储的盘符，通过 fsmcli 命令查看共享存储对应的盘符，见后面命令
/home/.8257798_root_zhanji_1_202012240000：解压后的 extent 备份
/vms/hw235-1:恢复文件存放的路径，新建共享存储或本地存储，保证容量足够，不要放在原有坏的共享存储上，避免数据覆盖
8257798_root_zhanji_1_202012240000_new:恢复的文件名,不要和原有目录文件重名，否则覆盖
---
fsmcli showpool --name defaultPool_hdd
…
device name: /dev/dm-0
device path: /dev/disk/by-id/dm-name-360000000000000000e0000003b75836c
device naa: 360000000000000000e0000003b75836c

6.10 共享存储空间释放方法

6.10.1 更改虚拟机总线类型手动释放共享卷空间

举例实施前主机后台 df –h 看该共享卷剩余空间为 596G

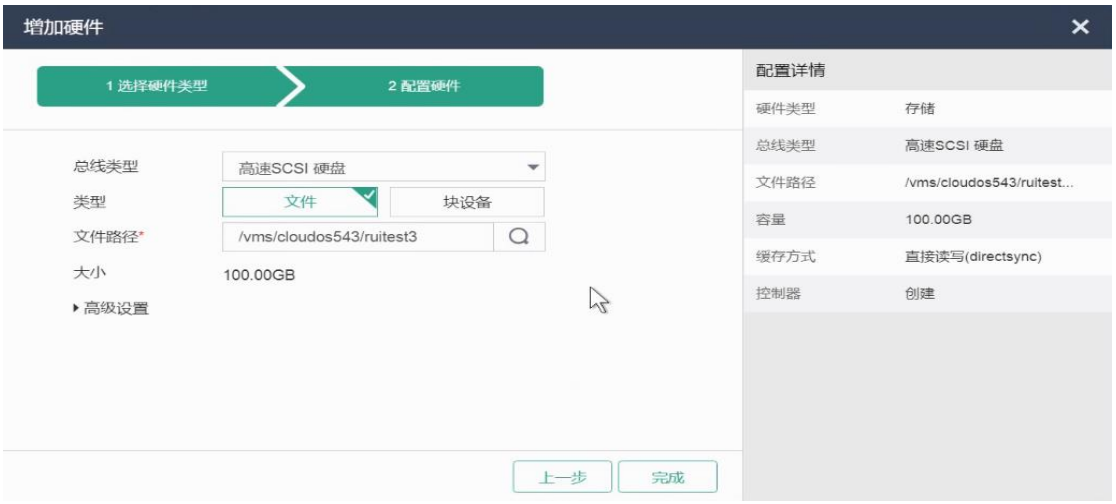
```
/dev/dm-18                                  6.0T  6.0T     0 100% /vms/UISblock
/dev/dm-19                                   16T   16T  596G  97% /vms/cloudos543
/dev/dm-17                                  3.9T  3.4T  503G  88% /vms/defaultPool_hdd
```

1. 虚拟机后台截图记录当前数据盘盘符及挂载路径

```
[root@localhost /]# df -h
文件系统                容量  已用  可用 已用% 挂载点
/dev/mapper/centos-root   50G  906M   50G    2% /
devtmpfs                 1.9G     0  1.9G    0% /dev
tmpfs                    1.9G     0  1.9G    0% /dev/shm
tmpfs                    1.9G  8.5M  1.9G    1% /run
tmpfs                    1.9G     0  1.9G    0% /sys/fs/cgroup
/dev/mapper/centos-home   26G   33M   26G    1% /home
/dev/vda1               1014M  143M  872M   15% /boot
tmpfs                    379M     0  379M    0% /run/user/0
/dev/vdb                  99G  4.3G   90G    5% /vms/ruitest
```

2. 关闭虚拟机后删除数据盘

3. 增加硬件重新挂上数据盘，并选择高速 SCSI 硬盘



4. 启动虚拟机后重新挂载数据盘，会发现盘符发生了改变，用新盘符重新挂载即可。如本例中 vdb 变为 sda

mount /dev/sda /vms/ruitest

```
[root@localhost ~]# lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
fd0               2:0    1    4K  0 disk
sda               8:0    0  100G  0 disk
sr0              11:0    1 1024M  0 rom
vda             252:0    0   80G  0 disk
├─vda1          252:1    0    1G  0 part /boot
└─vda2          252:2    0   79G  0 part
  ├─centos-root 253:0    0   50G  0 lvm  /
  ├─centos-swap 253:1    0  3.9G  0 lvm  [SWAP]
  └─centos-home 253:2    0 25.1G  0 lvm  /home
```

5. 虚拟机后台执行 fstrim /vms/ruitest 释放空间

```
[root@localhost ~]# df -h
文件系统                 容量  已用  可用 已用% 挂载点
/dev/mapper/centos-root   50G  906M   50G    2% /
devtmpfs                 1.9G     0  1.9G    0% /dev
tmpfs                    1.9G     0  1.9G    0% /dev/shm
tmpfs                    1.9G  8.5M  1.9G    1% /run
tmpfs                    1.9G     0  1.9G    0% /sys/fs/cgroup
/dev/vda1               1014M  143M  872M   15% /boot
/dev/mapper/centos-home   26G   33M   26G    1% /home
tmpfs                    379M     0  379M    0% /run/user/0
/dev/sda                  99G  4.3G   90G    5% /vms/ruitest
[root@localhost ~]# fstrim /vms/ruitest
[root@localhost ~]#
```

主机后台可以看到共享卷空间已由 596G 提升至 669G

```
/dev/dm-18                                 6.0T  6.0T     0 100% /vms/UISblock
/dev/dm-19                                  16T   16T  669G  96% /vms/cloudos543
/dev/dm-17                                 3.9T  3.4T  503G  88% /vms/defaultPool_hdd
```

6.10.2 删除文件自动释放共享卷空间方法

总线类型为高速 SCSI 硬盘的数据盘，虚拟机后台挂载数据盘时使用以下命令

mount -o discard /dev/sda /vms/ruitest

```
[root@localhost ~]# mount -o discard /dev/sda /vms/ruitest
[root@localhost ~]#
```

执行 mount 命令确认已为 discard

```
/dev/mapper/centos-home on /home type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
tmpfs on /run/user/0 type tmpfs (rw,nosuid,nodev,relatime,seclabel,size=387136k,mode=700)
/dev/sda on /vms/ruitest type ext4 (rw,relatime,seclabel,discard,data=ordered)
[root@localhost ~]#
```

主机后台查看共享卷剩余空间为 668G

```
/dev/dm-16                                500G  464G   37G  93% /vms/iso
/dev/dm-18                                6.0T  6.0T     0 100% /vms/UISblock
/dev/dm-19                                 16T   16T  668G  96% /vms/cloudos543
/dev/dm-17                                3.9T  3.4T  503G  88% /vms/defaultPool_hdd
```

```
[root@localhost ~]# df -h
文件系统                 容量  已用  可用 已用% 挂载点
/dev/mapper/centos-root   50G  906M   50G    2% /
devtmpfs                 1.9G     0  1.9G    0% /dev
tmpfs                    1.9G     0  1.9G    0% /dev/shm
tmpfs                    1.9G  8.5M  1.9G    1% /run
tmpfs                    1.9G     0  1.9G    0% /sys/fs/cgroup
/dev/vda1               1014M  143M  872M   15% /boot
/dev/mapper/centos-home   26G   33M   26G    1% /home
tmpfs                    379M     0  379M    0% /run/user/0
/dev/sda                  99G  4.3G   90G    5% /vms/ruitest
```

删除大文件后剩余空间自动提升至 672G

```
/dev/dm-16                                500G  464G   37G  93% /vms/iso
/dev/dm-18                                6.0T  6.0T     0 100% /vms/UISblock
/dev/dm-19                                 16T   16T  672G  96% /vms/cloudos543
/dev/dm-17                                3.9T  3.4T  503G  88% /vms/defaultPool_hdd
[root@UIS27 cloudos543]#
```

```
[root@localhost ~]# df -h
文件系统                 容量  已用  可用 已用% 挂载点
/dev/mapper/centos-root   50G  907M   50G    2% /
devtmpfs                 1.9G     0  1.9G    0% /dev
tmpfs                    1.9G     0  1.9G    0% /dev/shm
tmpfs                    1.9G  8.5M  1.9G    1% /run
tmpfs                    1.9G     0  1.9G    0% /sys/fs/cgroup
/dev/vda1               1014M  143M  872M   15% /boot
/dev/mapper/centos-home   26G   33M   26G    1% /home
tmpfs                    379M     0  379M    0% /run/user/0
/dev/sda                  99G   61M   94G    1% /vms/ruitest
```

6.11 SNMP相关

6.11.1 网管平台接收不到 get 响应

1. 问题定位

(1) snmp 服务开始监听 get port 配置端口后该端口被占用

使用 netstat -apn |grep xxxx 查看端口 xxxx(xxxx 为 get port 配置端口号)，查看该端口占用情况，可以看到最右侧的进程 pid，再使用 ps –aux | grep xxxxxx (xxxxxx 为进程 pid)命令，查看该端口被其他进程占用情况：

```
root@node111:~# netstat -apn |grep 12302
udp        0      0 0.0.0.0:12302           0.0.0.0:*                           1119905/nc
udp        0      0 0.0.0.0:12302           0.0.0.0:*                           1097438/python
root@node111:~# ps -aux | grep 1097438
root     1097438  0.3  0.7 401828 57108 ?        S    16:31   0:01 python /usr/bin/snmp-get-responder
root     1120413  0.0  0.0  10480  2144 pts/10   R+   16:39   0:00 grep --color=auto 1097438
root@node111:~# ps -aux | grep 1119905
root     1119905  0.3  0.0   9140   772 pts/5    S+   16:39   0:00 nc -lu 12302
root     1120757  0.0  0.0  10484  2220 pts/10   S+   16:39   0:00 grep --color=auto 1119905
```

除了 snmp-get-responder 进程外还有其余进程使用该端口，则认为此端口已被其余进程占用。

若确认此端口已被其余进程占用，需将其余进程关闭，或通过 kill xxxxxx (xxxxxx 为进程 pid)关闭其他进程。

```
root@node111:~# kill 1119905
root@node111:~# netstat -apn |grep 12302
udp        0      0 0.0.0.0:12302           0.0.0.0:*                           1097438/python
```

(2) 网管平台配置 snmp v1 版本 get 请求时配置 oid 错误

在存储端 leader 节点使用命令行 snmpget -v1 -c $community $ip:$port $oid，其中$community 为读团体名，不配置时输入 public，$ip 为存储端 ip，$port 为所配置的 get port 端口号，$oid 为网管平台所配置的 oid，如：snmpget -v1 -c public 172.16.156.111:12302 1.3.6.1.4.1.25504.1.7.1.12.0。可以查看存储端返回消息，如果为以下错误码，则说明 oid 配置错误。

```
root@node111:~# snmpget -v1 -c public 172.16.156.111:12302 1.3.6.1.4.1.25504.1.7.1.12.0
Error in packet
Reason: (noSuchName) There is no such variable name in this MIB.
Failed object: iso.3.6.1.4.1.25504.1.7.1.12.0
```

若确认 oid 配置错误，需检查 oid 修改为正确值，修改正确后，将返回如下格式“oid=string”信息。

```
root@node111:~# snmpget -v1 -c public 172.16.156.111:12302 1.3.6.1.4.1.25506.1.7.1.12.0
iso.3.6.1.4.1.25506.1.7.1.12.0 = STRING: "Failed to get the alarm content due to there is no alarm in current database"
```

```
root@node114:~# snmpget -v1 -c public 172.16.156.114:162 1.3.6.1.4.1.25506.1.7.1.12.0
iso.3.6.1.4.1.25506.1.7.1.12.0 = STRING: "'alarm_service_log': None, 'recovery_time': None, 'alarm_module': u'CLUSTER', 'alarm_time': '2018-02-06 14:54:57'
```

(3) 网管平台配置 snmp v2c 版本和 v3 版本 get 请求时配置 oid 错误

存储支持的 oid 范围如下：1.3.6.1.4.1.25506.1.7.1.2、1.3.6.1.4.1.25506.1.7.1.9、1.3.6.1.4.1.25506.1.7.1.10、1.3.6.1.4.1.25506.1.7.1.12、1.3.6.1.4.1.25506.1.7.1.13。在配置 get 请求时，网管平台配置的 oid 最后需要增加一位数字，该数字有效范围为 0 至 2147483647。查看 /var/log/onestor/snmp_get_responder.log 后台日志，如果有以下“NoSuchObjectError”错误提示，则说明 oid 配置错误，不在存储支持的 oid 范围内，mib 中不存在该 oid 节点。可能存在如下情况：oid 输入了多于正确 oid 的位数，如 1.3.6.1.4.1.25506.1.7.1.2.0.1，需检查位数是否正确。

```
2018-02-08 18:25:15,108 get_service_responder.py[line:48] ERROR: Failed to write the vars to respond the oid 1
.3.6.1.4.1.25506.1.7.1.100.10. Error type: <class 'pysnmp.smi.error.NoSuchObjectError'>,reason: NoSuchObjectEr
ror({'name': (1, 3, 6, 1, 4, 1, 25506, 1, 7, 1, 100, 10), 'idx': 0})
```

如果有以下“NoAccessError”错误，则说明 oid 配置错误，不在存储支持的 oid 范围内，mib 中该节点存在，但该节点无读写权限。可能存在如下情况：oid 输入了少于正确 oid 的位数，如 1.3.6.1.4.1.25506.1.7.1.2，需检查位数是否正确。

```
2018-02-11 02:22:57,469 get_service_responder.py[line:48] ERROR: Failed to write the vars to respond the oid 1
.3.6.1.4.1.25506.1.7.1.12. Error type: <class 'pysnmp.smi.error.NoAccessError'>,reason: NoAccessError({'name':
 (1, 3, 6, 1, 4, 1, 25506, 1, 7, 1, 12), 'idx': 0})
```

如果有以下“ValueConstraintError"，可能存在如下情况：oid 最后一位数字不在 0 至 2147483647 范围内，如 1.3.6.1.4.1.25506.1.7.1.2.2147483647,需检查最后一位数字是否在上述范围内

```
2018-02-11 02:33:50,187 get_service_responder.py[line:48] ERROR: Failed to write the vars to respond the oid 1
.3.6.1.4.1.25506.1.7.1.12.2147483648. Error type: <class 'pyasn1.type.error.ValueConstraintError'>,reason: <Co
nstraintsIntersection object at 0x7f52da432f90 consts <ValueRangeConstraint object at 0x7f52ef8f9a90 consts -2
147483648, 2147483647>, <ValueRangeConstraint object at 0x7f52da432f10 consts 0, 2147483647>> failed at: Value
ConstraintError('<ValueRangeConstraint object at 0x7f52ef8f9a90 consts -2147483648, 2147483647> failed at: Val
ueConstraintError(2147483648,)',) at Integer32
```

2. 处理方法

若确认 oid 配置错误，需检查 oid 修改为正确值，修改正确后，日志中将不会打印上述错误信息，提示“Success to write the vars”。

```
2018-02-11 02:22:52,535 get_service_responder.py[line:31] INFO: Start to process the response PDU. StateRefere
nce number:12540452
2018-02-11 02:22:52,535 get_service_responder.py[line:40] INFO: Start to get the content to respond. oid: 1.3.
6.1.4.1.25506.1.7.1.12.1
2018-02-11 02:22:52,587 get_service_responder.py[line:45] INFO: Success to write the vars. oid: 1.3.6.1.4.1.25
506.1.7.1.12.1, content: {'alarm_service_log': None, 'recovery_time': None, 'alarm_module': u'CLUSTER', 'alarm
_time': '2018-02-06 14:44:58', 'alarm_content': u'\u96c6\u7fa4\u5de5\u4f5c\u72b6\u51b5\u5f02\u5e38, \u72b6\u60
01\u4e3aHEALTH_WARN\u3002clock skew detected on mon.node115, mon.node116', 'alarm_status': u'un_recovery', 'co
nfirm_time': None, 'alarm_state': u'current', 'alarm_value': 1.0, 'alarm_key': u'0x020420018464421a-eb71-4d0f-
ba78-4195da0c3ee28464421a-eb71-4d0f-ba78-4195da0c3ee2cluster_health_warn', 'alarm_level': u'major', 'class_id'
: u'0x02042001', 'alarm_cause': u'1.\u96c6\u7fa4\u4e9a\u5065\u5eb7', 'alarm_recovery_tip': u'1.Handy\u9875\u97
62\u67e5\u770b\u96c6\u7fa4\u4e3b\u673a\u7ba1\u7406->\u76d1\u63a7\u8282\u70b9\u72b6\u6001\u662f\u5426\u6b63\u5e
38\u5426\uff0c\u767b\u9646mon\u8282\u70b9\u540e\u53f0\u67e5\u770b\u8fdb\u7a0b\u662f\u5426\u5b58\u5728\uff0c\u8
2e5\u4e0d\u5b58\u5728\u5219\u542f\u52a8\u8fdb\u7a0bstart ceph-mon-all\u3002\u542f\u52a8\u540e2\u5206\u949f\uff
0c\u518d\u6b21\u67e5\u770b\u8fdb\u7a0b\u662f\u5426\u5b58\u5728\uff0c\u82e5\u8fdb\u7a0b\u4e0d\u5b58\u5728\u5219
\u8054\u7cfbH3C\u5de5\u7a0b\u5e08\u3002\u662f\uff0c\u67e5\u770b\u96c6\u7fa4\u4e3b\u673a\u7ba1\u7406->\u5b58\u5
0a8\u8282\u70b9\uff0c\u786e\u8ba4\u662f\u5426\u67090SD \u5f02\u5e38\u3002\u5982\u679c\u67d0\u8282\u70b9\u4e0a\
u6240\u67090SD \u5747\u5f02\u5e38\uff0c\u8bf7\u68c0\u67e5\u8be5\u8282\u70b9\u7f51\u7edc\u73af\u5883\u548c\u914
d\u7f6e\uff0c\u5426\u5219\u8054\u7cfbH3C\u5de5\u7a0b\u5e08\u3002', 'index_id': 1L}
2018-02-11 02:22:52,590 get_service_responder.py[line:54] INFO: Success send the get response.StateReference n
umber:12540452
```

6.12 增值业务相关

6.12.1 业务查询详情结果与展示结果不一致

1. 问题定位

由于 handy 节点故障，导致增值业务在系统事件处理中更新数据库失败，从而表现为从数据库获取数据的展示界面结果与从内存获取数据的详情结果不一致。

2. 处理方法

- 如果是卷迁移特性，在界面将不一致的迁移删除，然后重新创建；
- 如果是卷拷贝特性，在界面停止不一致的拷贝，然后重新启动。

6.12.2 卷挂载给 windows 客户端在线创建快照可能会出现数据不一致情况

1. 问题定位

产品提供快照功能是存储侧快照，创建快照的瞬间无法保证主机测没有缓存数据，通 hang IO 实现多时间点同步，保证创建快照的时间点主机刷盘。存储侧快照创建时若 Windows 客户端有缓存机制，则无法保证创建的快照保护的数据与该时间点数据一致，可能是旧的数据。

2. 处理方法

主机侧需要 agent 软件配合进行快照创建时缓存刷盘。目前暂无此软件，可采用离线快照的方式规避数据不一致的问题。

6.12.3 同一个卷的不同时间点的多个只读快照或者可写快照同时映射给一个 windows 客户端，有些快照显示“没有初始化，未分配”，不可用

1. 问题定位

主机操作系统可能将快照卷和源卷识别为同一个卷。将源卷和快照卷映射给同一主机时，由于主机操作系统、卷管理等采用的卷识别机制（例如，Oracle ASM 应用场景中，主机通过 ASM 磁盘头信息识别不同的卷）可能会将源卷和快照卷识别为同一个卷，导致源卷和快照卷的数据被破坏。

2. 处理方法

建议不要将源 LUN 和快照 LUN 映射至同一主机。

6.12.4 对卷打快照后，handy 界面把卷移除映射后（不执行扫盘和断 iscsi 连接操作），进行快照回滚,原卷数据未恢复到快照时间点数据。

1. 问题定位

存储侧断开映射，主机侧未感知到映射已断开。主机侧保有数据缓存，存储侧进行快照回滚后，重新挂载给主机，主机缓存覆盖已回滚的卷数据。

2. 处理方法

进行快照回滚前进行下述操作任一即可。

(1) 解除映射并重新扫描硬盘。

(2) 断开 iscsi 连接。

6.12.5 原卷 mount 到目录下时，对原卷创建只读快照，创建完成后，只读快照不能 mount，提示 wrong fs type

1. 问题定位

Linux 客户端挂载原卷，新建的文件系统由于缓存的问题未能刷盘，此时创建存储侧快照，快照文件系统不完整，挂载时出现 super block 损坏错误。

2. 处理方法

Linux 客户端创建快照前进行解除挂载操作。

6.12.6 快照可能出现创建中，删除中和回滚中的中间状态

1. 问题定位

快照可能因为异常情况出现创建，删除，回滚失败的情况，并且由于集群异常，记录无法进行自动回退。

2. 处理方法

对于创建中和删除中创建的快照，可以进行手动删除操作清理残留记录；对于回滚中的快照记录，可以重新进行回滚操作。

6.13 兼容性问题

6.13.1 负载均衡在 intel ixgbe 网卡上导致存储访问慢的规避方案

通过命令 ethtool －i eth0 查看网卡 driver 是否为 ixgbe。

```
[root@onestor00 vm]# ethtool -i ens3f0
driver: ixgbe
version: 5.1.0-k
firmware-version: 0x800006db
expansion-rom-version:
bus-info: 0000:1a:00.0
supports-statistics: yes
supports-test: yes
supports-eeprom-access: yes
supports-register-dump: yes
supports-priv-flags: yes
```

通过命令 ethtool －k eth0 查看网卡的 LRO（large-receive-offload）功能是否关闭。

```
generic-segmentation-offload: on
generic-receive-offload: on
large-receive-offload: off
rx-vlan-offload: on
tx-vlan-offload: on
ntuple-filters: off
receive-hashing: on
highdma: on [fixed]
rx-vlan-filter: on
vlan-challenged: off [fixed]
tx-lockless: off [fixed]
netns-local: off [fixed]
tx-gso-robust: off [fixed]
```

通过命令 ethtool －K eth0 lro off 关闭 LRO 功能。

```
[root@onestor00 vm]# ethtool -K ens3f0 lro off
[root@onestor00 vm]# ethtool -k ens3f0
Features for ens3f0:
rx-checksumming: on
tx-checksumming: on
        tx-checksum-ipv4: off [fixed]
        tx-checksum-ip-generic: on
        tx-checksum-ipv6: off [fixed]
        tx-checksum-fcoe-crc: on [fixed]
        tx-checksum-sctp: on
scatter-gather: on
        tx-scatter-gather: on
        tx-scatter-gather-fraglist: off [fixed]
tcp-segmentation-offload: on
        tx-tcp-segmentation: on
        tx-tcp-ecn-segmentation: off [fixed]
        tx-tcp-mangleid-segmentation: off
        tx-tcp6-segmentation: on
udp-fragmentation-offload: off
generic-segmentation-offload: on
generic-receive-offload: on
large-receive-offload: off
```

在/etc/rc.local 文件中加入 ethtool －K eth0 lro off 命令，以便在重启的时候也能够生效。

6.13.2 低限制的 qos 策略引起客户端慢盘现象分析

1. 问题定位

客户端配置使用多个存储硬盘，如果在存储硬盘上关联了低带宽、低 IOPS 的 QoS 策略（相当于多个慢盘的场景），当每个存储硬盘的并发很大时（存储硬盘数 × 单个存储硬盘 IO 并发数 ＞ 启动器并发数），会概率出现 IO 跌 0 现象。IO 并发数参见本节第 2 条方法中的配置文件。

2. 处理方法

(1) 客户端进行压力分解。

客户端必须使用一个客户端多个硬盘的场景下，采用较大压力的使用方案，建议客户端安装多路径，且 iSCSI 连接进行多连接的配置方案，进行压力分解。

客户端侧不是必须使用一个客户端的场景下，采用较大压力的使用方案，建议使用多个客户端，将不同的存储硬盘挂载到不同的客户端上，进行压力分解。

(2) 客户端修改 iSCSI 启动器的 IO 限制。

在客户端修改 iSCSI 启动器配置文件，增大启动器的 IO 限制，方法如下：

打开 iSCSI 启动器的配置文件，默认路径为/etc/iscsi/iscsid.conf

找到配置文件中的 session and device queue depth 部分，将 node.session.cmds_max 从默认值修改为最大值 2048。

修改前为：

```
##################################
# session and device queue depth
##################################

# To control how many commands the session will queue set
# node.session.cmds_max to an integer between 2 and 2048 that is also
# a power of 2. The default is 128.
node.session.cmds_max = 128

# To control the device's queue depth set node.session.queue_depth
# to a value between 1 and 1024. The default is 32.
node.session.queue_depth = 32
```

修改后为：

```
##################################
# session and device queue depth
##################################

# To control how many commands the session will queue set
# node.session.cmds_max to an integer between 2 and 2048 that is also
# a power of 2. The default is 128.
node.session.cmds_max = 2048

# To control the device's queue depth set node.session.queue_depth
# to a value between 1 and 1024. The default is 32.
node.session.queue_depth = 32
```

修改完成后，进行启动器的重启。

6.14 虚拟机添加加密密狗无法识别

部分厂商的加密狗不支持网络 USB 方式，在使用前需要先进行对接测试。

如果遇到问题，请联系 H3C 技服人员处理。

6.14.1 USB 插到 cvk 主机上后，主机无法识别到该设备

1. 问题现象

把 USB 设备插到 cvk 主机后，在 UIS 的 WEB 管理界面给虚机添加 USB 设备时发现找不到该设备。

2. 问题定位

(1) USB 插槽可能没有插对，USB 设备换一个插槽试试。用小辫子的，可以尝试把 USB 设备直接插到服务器内部的 USB 插槽上。若服务器有多个类型的 USB 插槽，应把相应型号的 USB 插到对应插槽上。可用 lsusb –t 检查 USB 设备插的插槽是否正确。例：

```
 root@cvk-163:~# lsusb -t
/:  Bus 04.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/6p, 5000M
/:  Bus 03.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/15p, 480M
/:  Bus 02.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
    |__ Port 1: Dev 2, If 0, Class=hub, Driver=hub/8p, 480M
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
|__ Port 1: Dev 2, If 0, Class=hub, Driver=hub/6p, 480M
```

命令返回结果里的 **UHCI** 表示 USB1.1，**EHCI** 表示 USB2.0，**XHCI** 表示 USB3.0。一般 USB1.1 最大传输速率为 12Mbps，USB2.0 最大传输速率为 480Mbps，USB3.0 最大传输速率为 5Gbps。

如果服务器支持多种 USB 总线标准，给服务器添加一个 USB2.0 设备后，在 USB2.0（ehci-pci）的总线下新增一个 USB 设备，则说明 USB 设备插的插槽是正确的。

(2) 目前 USB 设备有 3.0、2.0、1.0，虽然是向下兼容，但一些 USB 设备兼容性并不好，例如 USB Key 一般为 1.0 设备，插到只有 USB3.0 插槽的服务器上时，建议在服务器 bois 里把 USB3.0 禁用。

(3) 进行上述操作后，还无法识别到，继续往下排查。在 cvk 后台，拔下插上 USB 设备前后，用 lsusb 命令查看是否有新增设备，例如：

```
root@ CVK:~# lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 005 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 004 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 006 Device 002: ID 03f0:7029 Hewlett-Packard
Bus 006 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

- 如果没有，说明 Ubuntu 系统没有识别到该设备，Linux 系统可以支持市面上绝大多数 USB 设备，USB 设备本身可能有问题，尝试把 USB 设备插到办公 PC 上检查该 USB 设备是否可正常运行，如果可以正常运行，说明 USB 设备本身是正常的。用以下方法排查是 CAS 系统问题还是服务器不兼容该 USB 设备：
  a. 服务器裸机安装办公 PC 一样的系统，插上 USB 设备检查系统内是否可以识别。
  - 若不可以识别，说明是服务器不兼容该 USB。
  - 若可以识别到，说明服务器没有问题，是支持该 USB 设备的。

  b. 服务器裸机安装原生 Centos 系统，插上 USB 设备检查是否可以识别。
  - 若不可以识别，说明 Centos 系统本身缺少对该设备的支持。UIS 是基于 Centos 系统的，所以也不支持该 USB 设备。
- 如果有，说明 Centos 系统已识别到该设备，继续下面步骤排查。

(4) 用 virsh nodedev-list usb_device 查看新增 USB 设备的设备名称，例如：

```
root@ CVK:~# virsh nodedev-list usb_device
usb_2_1_5
usb_usb1
usb_usb2
usb_usb3
usb_usb4
```

新增 usb 设备名称为：usb_2_1_5；

(5) 用 virsh nodedev-dumpxml xxx 查看新增 USB 设备的 XML 信息，（xxx 为用 virsh nodedev-list usb_device 看到的新增 usb 的设备，如上面的 usb_2_1_5），例：

```
root@CVK:~# virsh nodedev-dumpxml usb_2_1_5
<device>
  <name>usb_2_1_5</name>
  <path>/sys/devices/pci0000:00/0000:00:1d.0/usb2/2-1/2-1.5</path>
  <parent>usb_2_1</parent>
```

```
  <driver>
    <name>usb</name>
  </driver>
  <capability type='usb_device'>
    <bus>2</bus>
    <device>70</device>
    <product id='0x6545'>DataTraveler G2 </product>
    <vendor id='0x0930'>Kingston</vendor>
  </capability>
</device>
```

查看 bus ID、device ID、product ID、vendor ID 是否正常，有没有为空的情况。如果这些都显示正常，但是 web 页面还是找不到 USB 设备，请联系 UIS 开发人员。

6.14.2 USB 设备加载给虚机后，虚机内部看到在设备管理器无法识别到该设备，或一闪消失不见，或设备上显示有感叹号。

1. 问题现象：

把 USB 设备加载给虚机后，在虚机内部设备管理器，找不到新增的 USB 设备，或者新增 USB 设备一闪消失不见，或者看到新增 USB 设备显示有感叹号。

2. 问题分析：

(1) USB 插槽可能没有插对，USB 设备换一个插槽试试。用小辫子的，可以尝试把 USB 设备直接插到服务器内部的 USB 插槽上。若服务器有多个类型的 USB 插槽，应把相应型号的 USB 插到对应插槽上。可用 lsusb –t 检查 USB 设备插的插槽是否正确。例：

```
root@cvk-163:~# lsusb -t
/:  Bus 04.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/6p, 5000M
/:  Bus 03.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/15p, 480M
/:  Bus 02.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
    |__ Port 1: Dev 2, If 0, Class=hub, Driver=hub/8p, 480M
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
|__ Port 1: Dev 2, If 0, Class=hub, Driver=hub/6p, 480M
```

UHCI 表示 USB1.1，**EHCI** 表示 USB2.0，**XHCI** 表示 USB3.0。一般 USB1.1 最大传输速率为 12Mbps，USB2.0 最大传输速率为 480Mbps，USB3.0 最大传输速率为 5Gbps。

例如服务器支持多种 USB 总线标准，给服务器添加一个 USB2.0 设备后，在 USB2.0(ehci-pci)的总线下新增一个 USB 设备，则说明 USB 设备插的插槽是正确的。

- USB 设备若是 USB Key、加密狗、短信猫，这些设备一般是 USB1.0 的，而服务器又只有 USB3.0 插槽，建议在 bois 里把 USB3.0 禁用。
- 检查 CVK 主机是否可以正常识别到该 USB 设备，拔下插上 USB 设备，用 virsh nodedev-list usb_device 检查是否有新增 usb 设备
  - 若没有，按上面 6.14.1 USB 插到 cvk 主机上后，主机无法识别到该设备问题排查方法排查。
  - 若有，继续下面步骤排查。
- 检查给虚机加载该设备时，选择的 USB 控制器是否正确，确定该设备 USB 型号，是 USB1.0，USB2.0，还是 USB3.0？给虚机添加 USB 设备时选择正确的 USB 控制器。一般 USB Key、加密狗、短信猫选择使用 USB1.0 控制器。

(2) 考虑是否驱动有问题，驱动版本和虚机操作系统是否匹配。务必确保驱动是正确的，可在物理机上装相同操作系统测试该驱动是否正常，或联系 USB 设备厂商咨询。也可在 vmawre 平台创建同样的虚机，装上此驱动加载 USB 设备看虚机内部是否可用。确保驱动是正确的情况下，虚机内部识别该设备还是有问题，继续下面步骤排查。

(3) 用 virsh nodedev-dumpxml xxx 查看新增 USB 设备的 XML 信息（xxx 为用 virsh nodedev-list usb_device 看到的新增 usb 的设备），例如：

```
root@ CVK:~# virsh nodedev-list usb_device
usb_2_1_5
usb_usb1
usb_usb2
usb_usb3
usb_usb4
新增 usb 设备名称为：usb_2_1_5；
  root@CVK:~# virsh nodedev-dumpxml usb_2_1_5
<device>
  <name>usb_2_1_5</name>
  <path>/sys/devices/pci0000:00/0000:00:1d.0/usb2/2-1/2-1.5</path>
  <parent>usb_2_1</parent>
  <driver>
    <name>usb</name>
  </driver>
  <capability type='usb_device'>
    <bus>2</bus>
    <device>70</device>
    <product id='0x6545'>DataTraveler G2 </product>
    <vendor id='0x0930'>Kingston</vendor>
  </capability>
</device>
```

把 usb 设备加载给虚机后，再次用 virsh nodedev-dumpxml xxx 查看新增 USB 设备的 XML 信息，观察 device id、product id、vendor id 对应的值是否有变化。若 device id、product id、vendor id 对应的值在设备加载给虚机后有变化。考虑可能是服务器和 USB 不兼容导致，在保证 USB 设备可用的情况下，用该服务器裸机装虚机使用的操作系统，观察是否可以正常使用该 USB 设备，系统内部日志是否有报错。需检查 USB 设备是可用的，不仅仅是能看到设备。若服务器裸机装虚机系统后，可以正常使用 USB 设备，请联系 UIS 开发人员。

6.14.3 USB3.0 使用问题

USB3.0 的设备，给虚机添加 USB 设备时在 web 界面上选择控制器为 USB3.0，加载后若在虚机内部找不到新加载的设备，可能原因：

(1) 虚机内部缺少 USB3.0 驱动，曾处理过虚机系统 Windows Server 2008 R2 Enterprise，给虚机添加 USB3.0 移动硬盘，虚机内部无法识别的问题，最终定位是由于 Windows Server 2008 R2 Enterprise 系统内部没有自带 USB3.0 驱动。USB3.0 是比较新的协议，部分老的操作系统没有自带对应驱动，需要下载。

系统支持 USB3.0 的可以在设备管理器内查看到如下红框中所示内容。

(2) USB3.0 设备和服务器的兼容，把 USB3.0 设备插到装有 UIS 的服务器上后，ssh 终端登录，用 lsusb –t 命令查看不到新增设备。

6.14.4 USB 转串口设备使用问题

USB 转串口设备插到装有 UIS 的服务器上后，ssh 终端登录，用 lsusb –t 命令查看是否有新增 usb 设备，查看新增设备的速度，速度若为 12Mbps，在给虚机添加 USB 设备时选择 USB1.0 的控制器；速度为 480Mbps，在给虚机添加 USB 设备时选择 USB2.0 控制器。

曾处理过的问题：

(1) USB 转串口设备，插到装有 UIS 的服务器上后，ssh 终端登录，用 lsusb –t 命令查看，可以看到新增设备，但是加载给虚机后，在虚机内部无法看到新增的串口设备，在虚机内部安装 USB 转串口驱动后，仍无法在虚机内部看到新增串口设备，最终定位是由于在给虚机添加 USB 设备时，选择的控制器为 USB2.0 速率不匹配导致的，改成 USB1.0 控制器后，虚机内部可以看到新增的串口设备了。

(2) 四个 USB 转串口线一端分别连接四个交换机，另一端同时插到一个装有 UIS 的服务器上，ssh 终端登录，用 lsusb –t 命令查看，不能同时看到四个新增设备。反复的插拔，有时能看到一个，有时能看到两个，有时能看到三个。当插上不能识别的 USB 接口时，syslog 有如下日志打印：

```
Jun  1 11:25:49 dycvm01 kernel: [63112.487196] usb 1-1.5: new full-speed USB device number 26 using ehci-pci
Jun  1 11:25:49 dycvm01 kernel: [63112.559148] usb 1-1.5: device descriptor read/64, error -32
Jun  1 11:25:49 dycvm01 kernel: [63112.735043] usb 1-1.5: device descriptor read/64, error -32
Jun  1 11:25:49 dycvm01 kernel: [63112.910955] usb 1-1.5: new full-speed USB device number 27 using ehci-pci
Jun  1 11:25:49 dycvm01 kernel: [63112.982912] usb 1-1.5: device descriptor read/64, error -32
Jun  1 11:25:49 dycvm01 kernel: [63113.158804] usb 1-1.5: device descriptor read/64, error -32
Jun  1 11:25:49 dycvm01 snmpd[5103]: Connection from UDP: [10.166.6.162]:53810->[10.166.6.7]
Jun  1 11:25:49 dycvm01 kernel: [63113.334712] usb 1-1.5: new full-speed USB device number 28 using ehci-pci
Jun  1 11:25:50 dycvm01 kernel: [63113.742336] usb 1-1.5: device not accepting address 28, error -32
Jun  1 11:25:50 dycvm01 kernel: [63113.814462] usb 1-1.5: new full-speed USB device number 29 using ehci-pci
Jun  1 11:25:50 dycvm01 kernel: [63114.222102] usb 1-1.5: device not accepting address 29, error -32
Jun  1 11:25:50 dycvm01 kernel: [63114.222220] hub 1-1:1.0: unable to enumerate USB device on port
```

出现这种日志是 USB 设备在与服务器建立连接时和总线协商有问题，建议排查该服务器是否兼容这种使用方式，最终定位是服务器不兼容，现场用的 HP FlexServer R390 服务器，换成 R590 的服务器后，可以正常识别出新增的四个设备。

6.15 性能提升

6.15.1 磁盘性能优化

E0705 和 E0706 版本的磁盘队列模式是 cfq 模式（E0707 版本），这个模式下 ssd 性能很差，而且 OCFS2 共享存储卷的 IO 性能也很差，最终导致集群性能差，引起虚拟机性能差，需要改成 deadline 模式

(1) 永久修改

```
 [root@cvknode1 ~]# cat /proc/cmdline
BOOT_IMAGE=/boot/vmlinuz-4.14.0-generic root=UUID=da51eb22-6c64-4b3b-af57-960a117823c4 ro biosdevname=0 rhgb elevator=deadline transparent_hugepage=always net.ifnames=0 crashkernel=256M quiet
```

修改 grub 配置：

```
python /opt/bin/util_kernel_cmdline.pyc -s elevator=deadline transparent_hugepage=always net.ifnames=0 crashkernel=256M
```

如果有其他额外的 grub 配置也需要作为这个命令的参数。

(2) 在线修改方式：

修改 sd 设备：

```
for i in `ls /sys/block/sd*/queue/scheduler`; do echo "deadline" > ${i};done
```

修改 dm 设备：

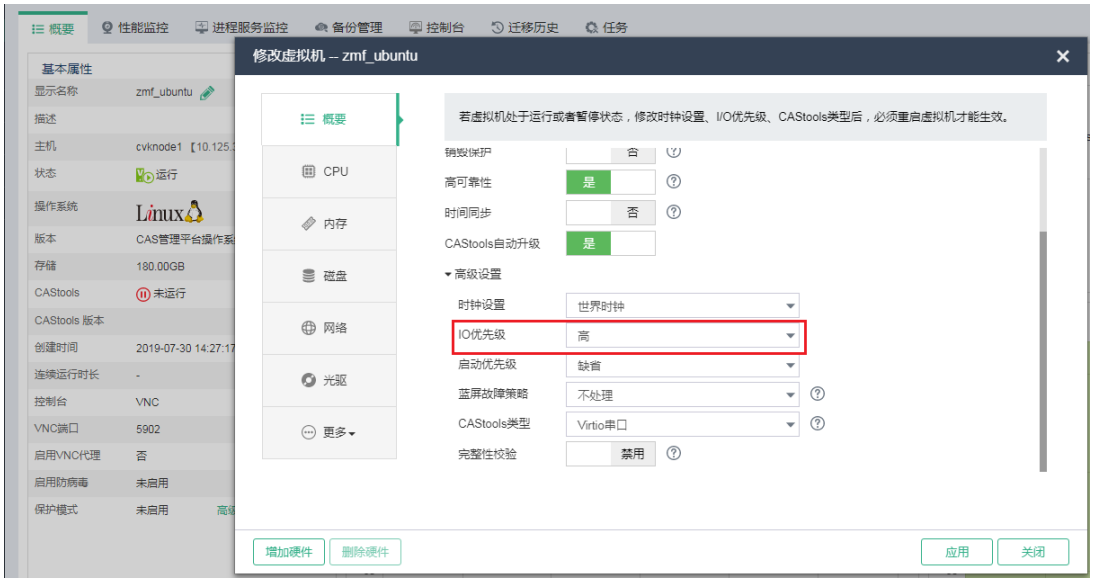
```
for i in `ls /sys/block/dm*/queue/scheduler`; do echo "deadline" > ${i};done
```

永久修改方式需要重启主机才能生效，在线修改方式对于新增加的块设备会失效，还是默认的 cfq 方式。

6.15.2 性能优化

1. 调整 IO 优先级

在[虚拟机/修改虚拟机/概要]页签修改 IO 优先级为“高”。



2. 调整 CPU 工作模式”

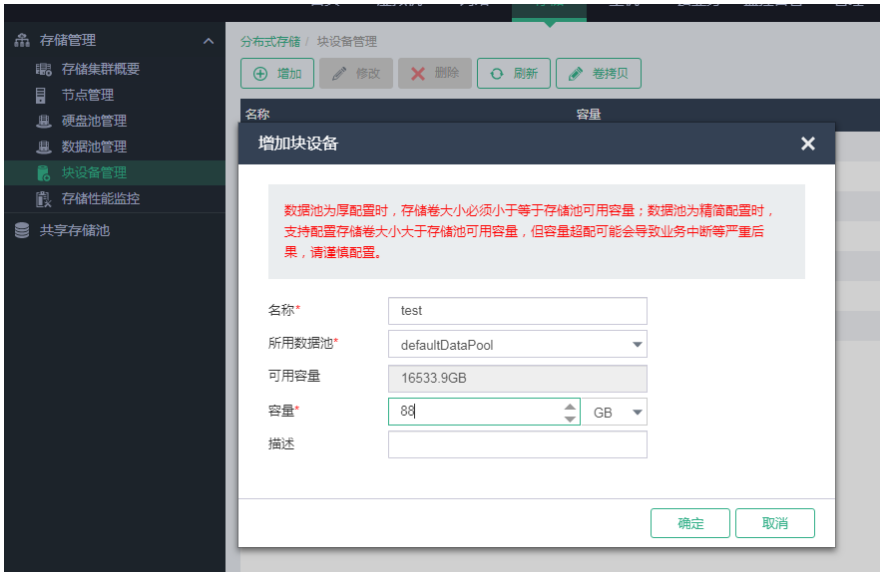
在[虚拟机/修改虚拟机/CPU]页面修改 CPU 工作模式为“直通模式”。

兼容性模式是默认模式，该种模式下虚拟化内核软件模式的通用标准虚拟 CPU，有点是兼容性好；直通模式则将物理服务器 CPU 型号和大部分功能透传给虚拟机，能够提供最优的性能。

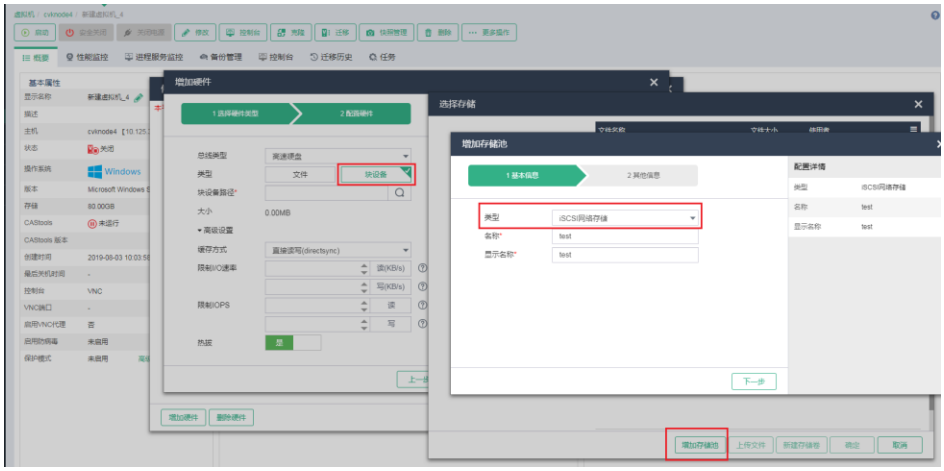
3. 虚拟机磁盘挂载方式调整

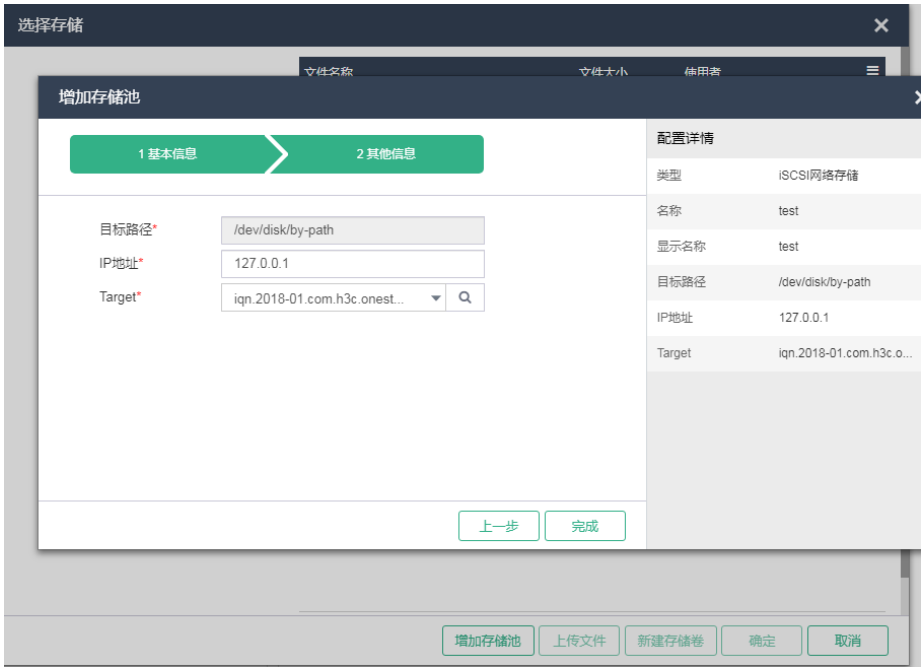
有共享存储的情况下，创建虚拟机的时候默认情况下是在共享存储划一个卷用于虚拟机的硬盘，针对对性能要求比较高的场景，可以使用裸块的方式将卷直接给虚拟机使用，跳过了 cvk 的文件系统一层。

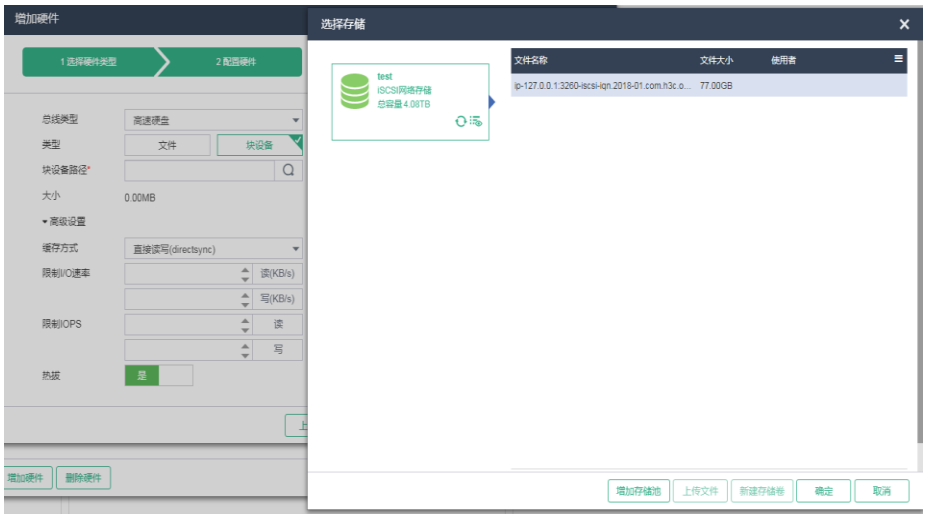
(1)　创建一个卷用于虚拟机磁盘



(2)　通过裸块方式挂载给虚拟机



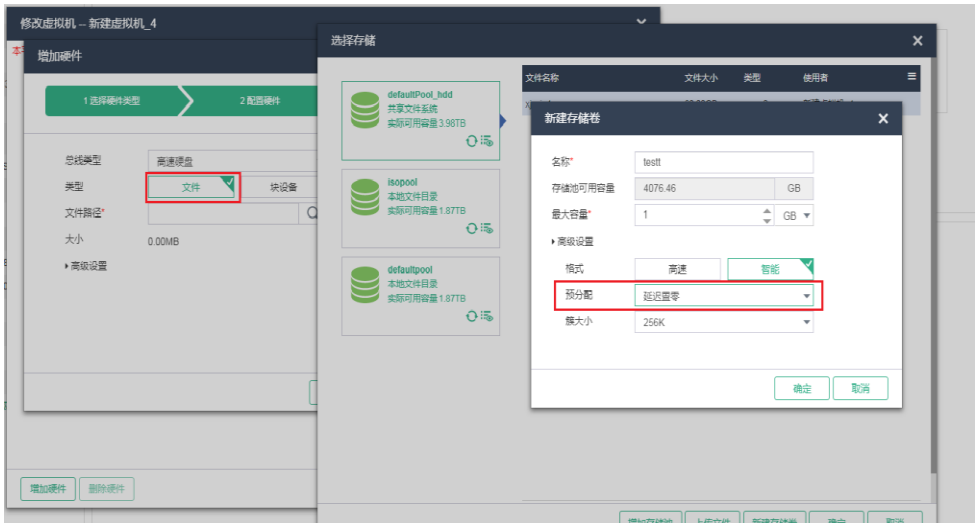




通过裸块方式挂载的虚拟机磁盘信息如下：

4. 虚拟机磁盘预配置方式调整

(1) 针对必现部署在共享存储的虚拟机，可以通过调整磁盘预配置方式提高性能，在创建卷的时候，预配置方式改成“延迟置零”。



(2) 增加虚拟机内存大小。

(3) 日志级别调整，登录任意一台 CVK 后台，执行如下命令。

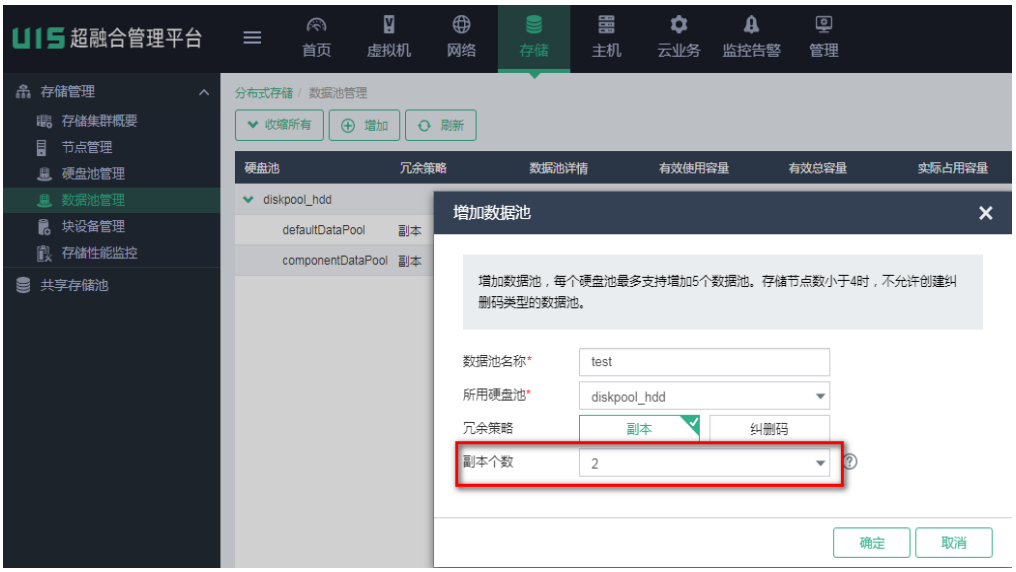
```
ceph tell osd.* injectargs --debug_osd=1/1
ceph tell osd.* injectargs --debug_ms=0/0
ceph tell osd.* injectargs --debug_bluestore=1/1
ceph tell osd.* injectargs --debug_bluefs=1/1
ceph tell osd.* injectargs --debug_rocksdb=1/1
ceph tell osd.* injectargs --debug_bdev=1/1
```

(4) IO Size 调整

登录所有 cvk 后台，所有节点都要执行如下命令，该调整仅适用于小 IO，如数据库，对于拷贝修改意义不大。

```
cd /proc/sys/dev/flashcache;for i in `ls`; do cd ${i}; echo 16 > skip_seq_thresh_kb; cd ..;
done  //16 表示对大于该值的 io，跳过 flashcache
```

(5) 副本数更改

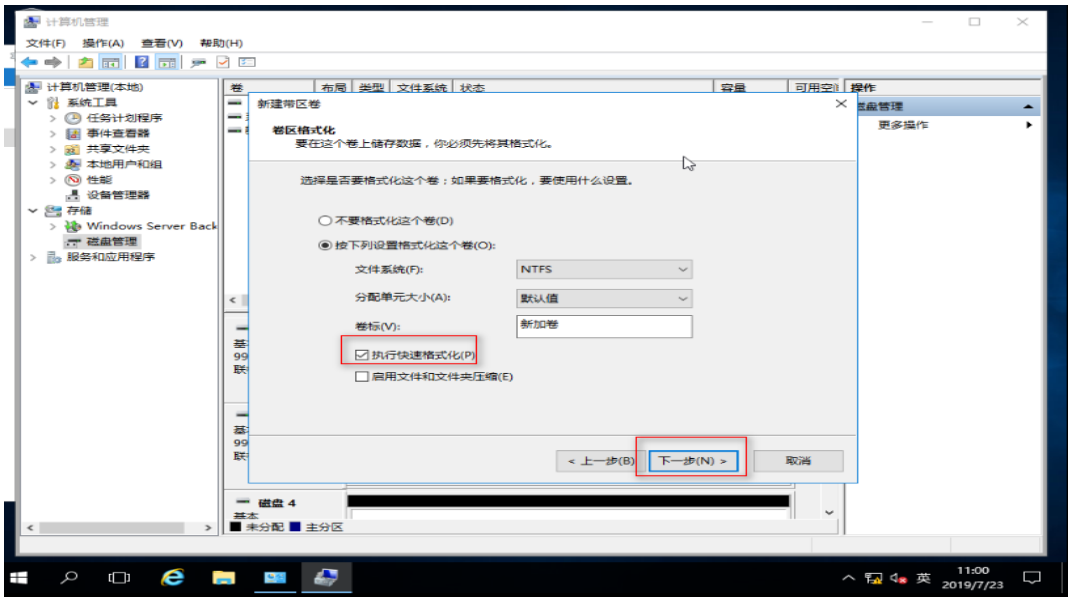


通过调整副本数可以起到提供性能的目的。

需要注意的是调整副本数会触发数据平衡，存在一定风险，如果要修改请联系研发进行操作。

(6) Window 带区卷

在添加给虚拟机的磁盘后，进行格式化操作时，请勾选“执行快速格式化”



6.16 操作系统及虚拟机修复

注意

- 此文提供普通的 Linux 和 Windows 修复处理过程，其他系统可参考进行。
- 容灾系统修复，存在不能确保完全成功的风险，请预先考虑并做好备份等。
- 修复方法并非百分百能够将虚拟机修复，如损坏严重，无法采用 ISO 或相关工具修复，则需要考虑用专业的容灾修复工具进行数据的恢复和抢救：比如 Diskgenius，diskrec 等。必要时需要客户联系专业的数据恢复公司协助。

6.16.1 修复前的准备

1. 系统磁盘的备份

对于损坏系统的硬盘，推荐预先进行整盘的备份，以防一次修复失败，可以尝试更多的修复方式。

对于损坏的硬盘，可以采用 dd 或其他备份工具，将磁盘复制备份。

虚拟化系统下，可将虚拟机的镜像文件备份，clone 到另外的存储池。

或者在磁盘数据所在的存储侧做快照，以防修复的意外情况。

2. 准备好对应的 ISO 系统

Linux 系统，准备好一个 CentOS 或 Ubuntu 的 ISO 安装盘，以便对于 Linux 系统目录进行修复处理。

Windows 系统推荐和损坏系统相同版本 ISO 文件，或光盘。

注意

- 推荐采用和系统相同的版本 ISO，或者更加新的版本 ISO 进行挂载修复；
- 在修复过程中发现，旧版的 ISO 中的文件系统的格式可能和新的不兼容，从而导致修复失败。

6.16.2 Linux 损坏系统的修复处理步骤

1. 为待修复系统的挂上光驱，并配置通过光驱引导，重启系统

CAS 虚拟化环境下给待修复的虚拟机挂上 ISO 版本光驱，“修改虚拟机”界面中修改引导顺序，优先从光驱引导。

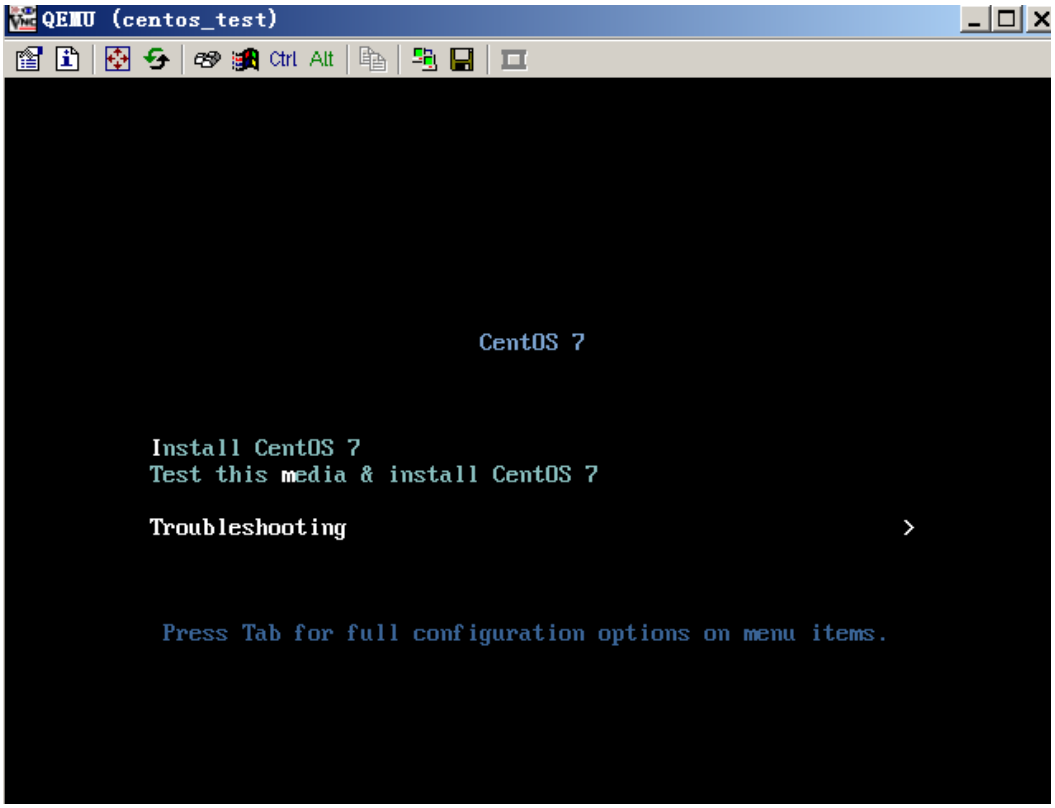
2. 启动系统，在终端上进行尝试修复；

虚拟化环境，在 CAS 界面上找到这台虚拟机使用的 cvk 的 ip 名称和对应的 vnc 的端口，通过 pc 端安装的 vnc client 连接端口； 推荐采用 tight VNC 等 PC 端连接。

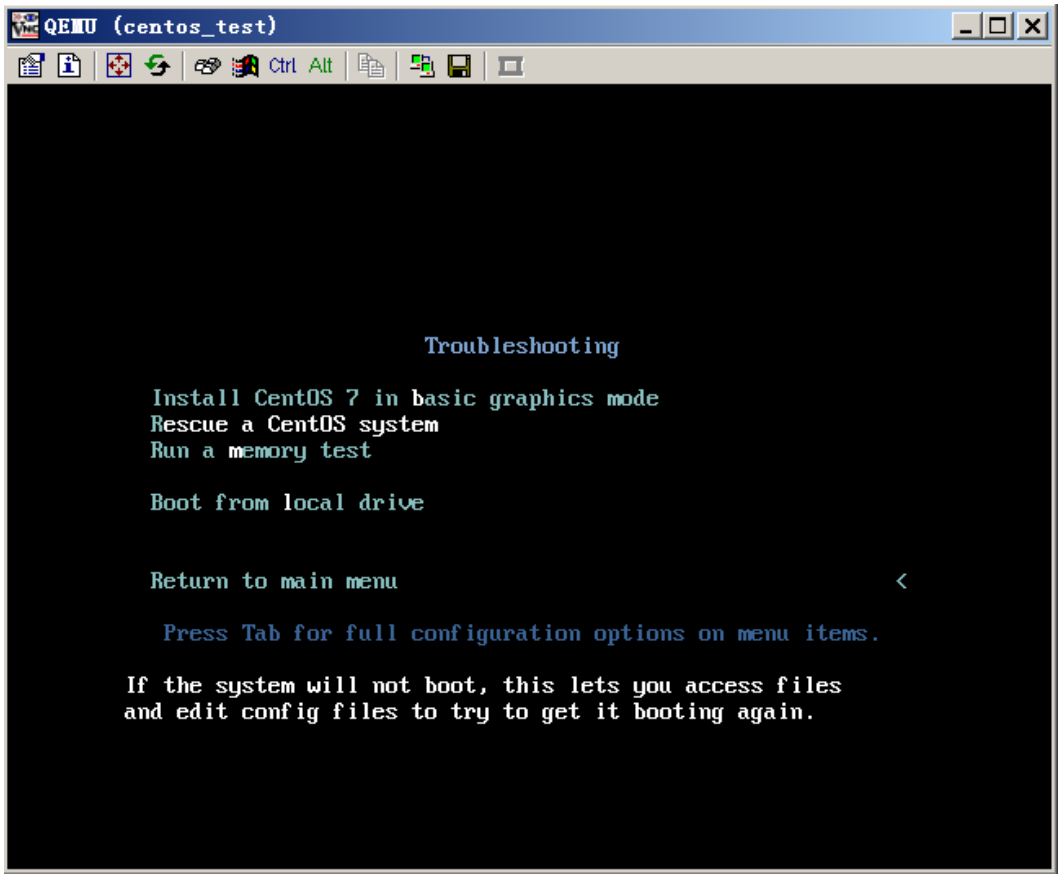
注意

不推荐采用浏览器的控制台，发现部分浏览器，操作几次后，有可能需要频繁情空浏览器的缓存才能打开对应的页面

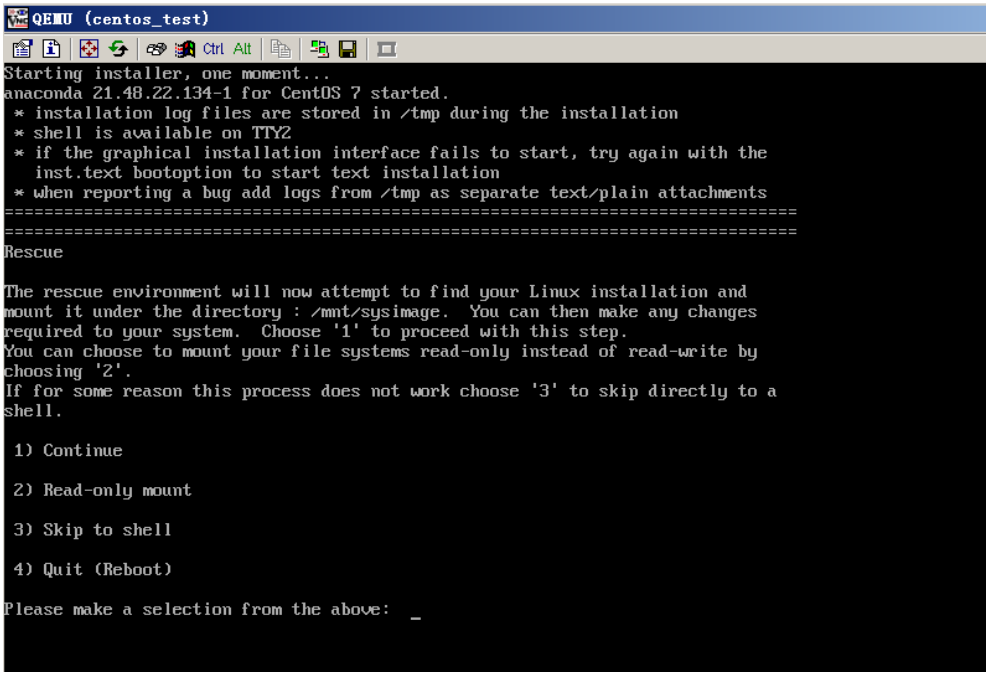
3. 在控制界面上 centos 选择 Trouble Shoot



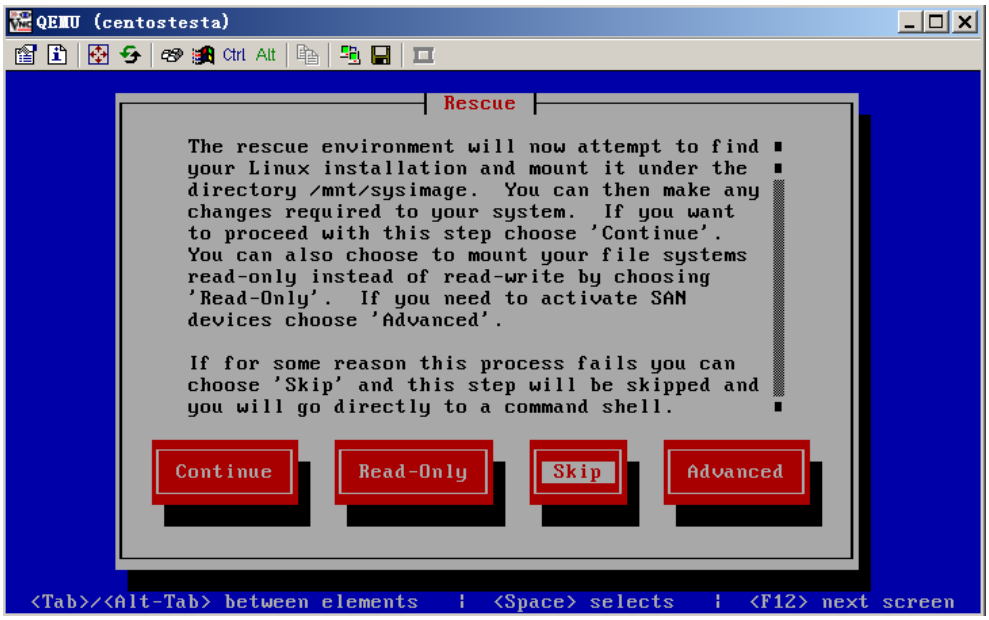
4. 选择 Rescue a CentOS System



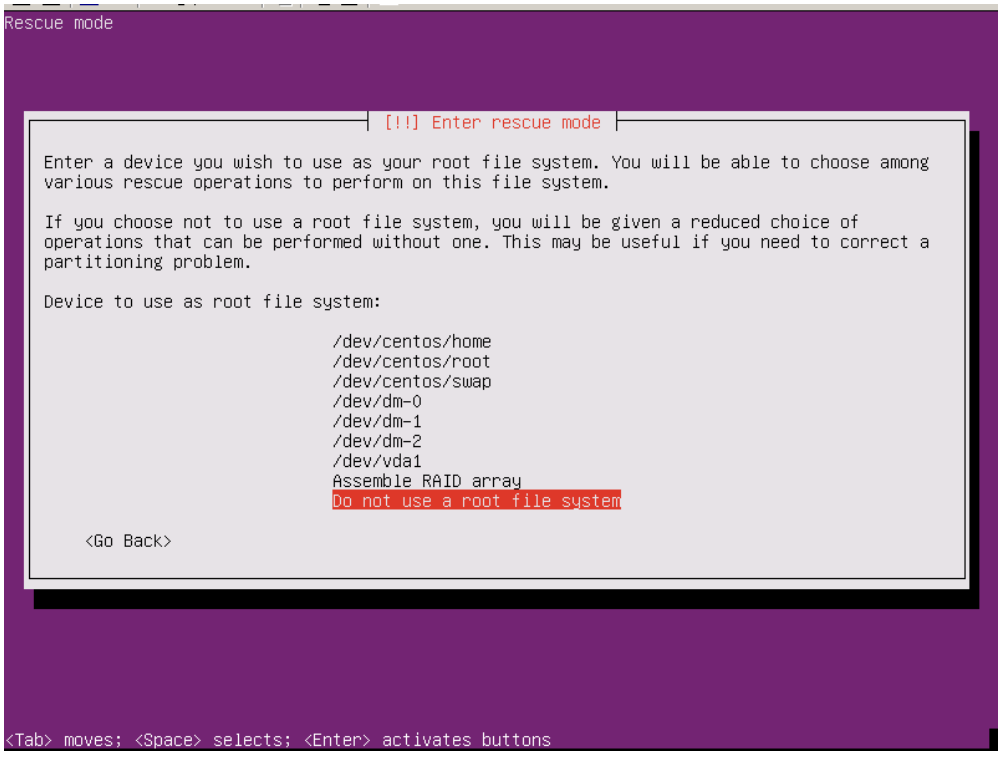
5. 选择 3，进入 shell 命令

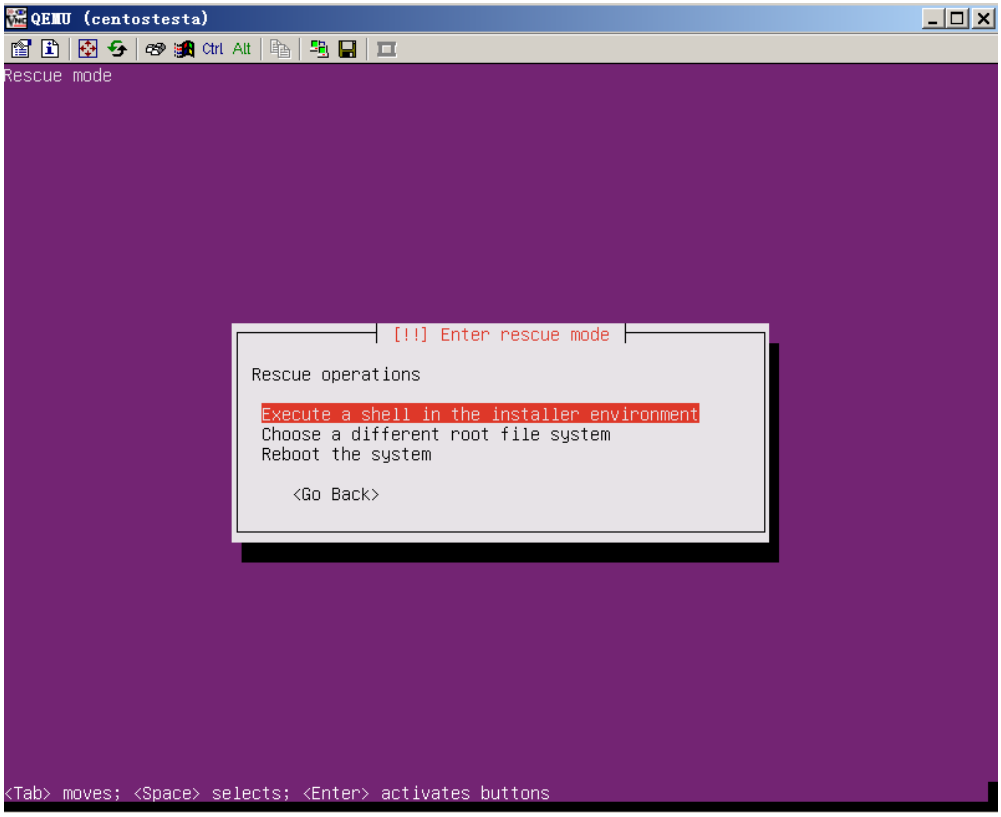


如果采用了旧版本的 Centos ISO，可以选择对应的“Skip”按钮，进入到 shell 界面。旧版 Centos 有“Continue”、“Read- only”、“Skip”和“Advanced”。



如果是 Ubuntu ISO 进行修复，则选择“Execute a shell in the installer environment”。





⚠ 注意

- 发现 Ubuntu1804 ISO 修复模式没有默认加载 xfs 的相关工具；推荐 xfs 的修复采用 centos 的最新版本进行；
- ISO 的选择匹配的，或更新版本的 ISO。

---

6. 采用 lvs 查看是否使用 lv

如下图，查询出 3 个 lv，swap 可以不用修复; 对应的 vg 的名称是 centos。

```
bash-4.1# lvs
  LV   VG     Attr       LSize  Pool Origin Data%  Meta%  Move Log Cpy%Sync Conv
ert
  home centos -wi------- 23.33g

  root centos -wi------- 47.79g

  swap centos -wi-------  7.88g
```

采用 lv 命令激活对应的 lv，使其可读。

```
lvchange -a y centos/home
lvchange -a y centos/root
```

检查下对应的 lv 上的文件系统，不同的文件系统需要采用不同的修复命令进行。

```
blkid /dev/centos/home
```

```
bash-4.1# lvchange -a y centos/home
bash-4.1# lvchange -a y centos/root
bash-4.1# blkid /dev/centos/home
/dev/centos/home: UUID="45b5a791-4e45-4d61-bf41-108ea3e1fdc5" TYPE="xfs"
bash-4.1# blkid /dev/centos/root
/dev/centos/root: UUID="dd4e68fc-e687-4b56-a06f-aa5b7d3093d3" TYPE="xfs"
```

注意

- 不同的安装系统的 vg 可能不同，有的是 centos，有的是 VolGroup01 等，需要根据实际的输出内容进行合适的选择和判断；
- 如果系统没有使用到 lvm，则可以采用 blkid 查询出对应的磁盘分区的文件系统，需要对应的 /dev/sdaX 的分区上文件系统即可。

7. 对于 xfs 的修复

```
xfs_repair /dev/centos/lv_root
```

如果存在日志系统，修复失败，此时请联系研发人员。

8. Ext4 的修复

```
fsck /dev/datavg/lv_data
```

中间有可能会要求输入 yes，请输入即可；其它文件系统的修复步骤类似。

9. 关闭虚拟机：执行命令 init 0

10. 将系统挂在 iso 光驱卸载，修改为原来的从硬盘引导顺序，重新启动系统；

11. 启动系统，检查系统的业务是否正常

6.16.3 Windows 的修复操作和步骤

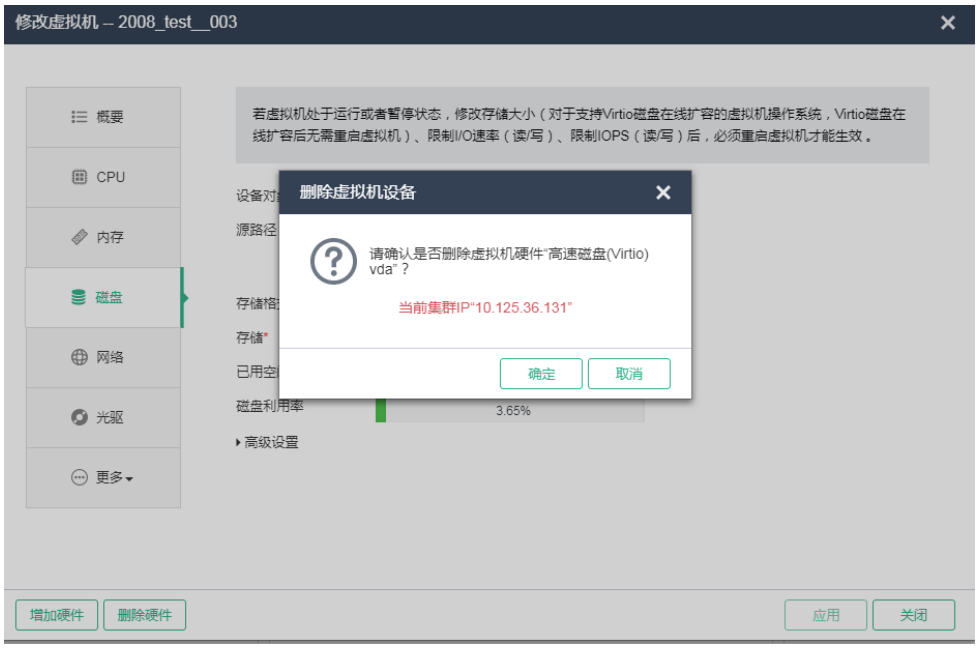
1. 错误现象

某局点一台 Windows 2008 虚拟机在 CAS 升级之后，启动虚拟机提示操作系统需要修复，选择修复一直卡在加载画面；选择正常启动则一直黑屏。

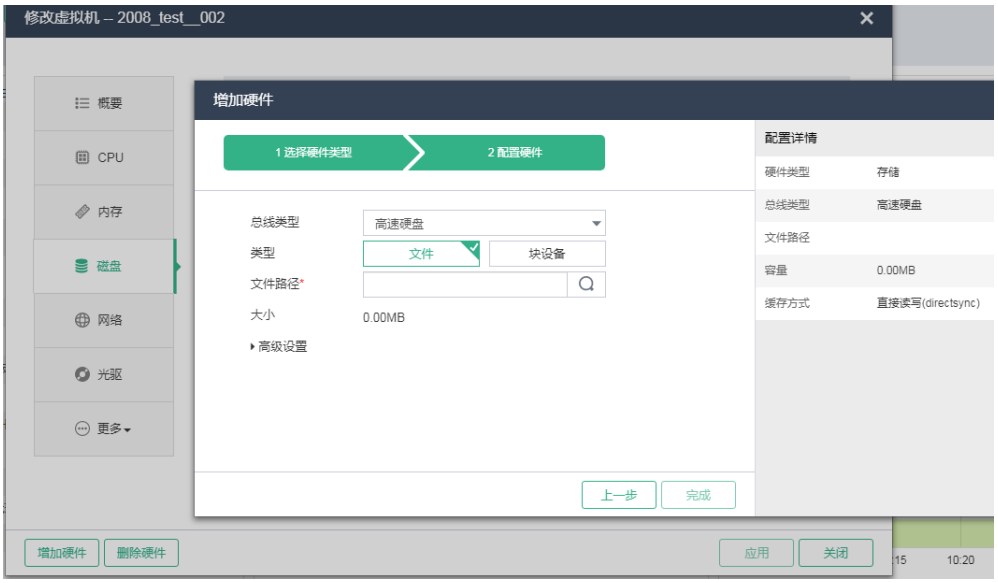
2. 修复步骤

(1) 将磁盘挂给另外一台正常的 Windows 虚拟机上

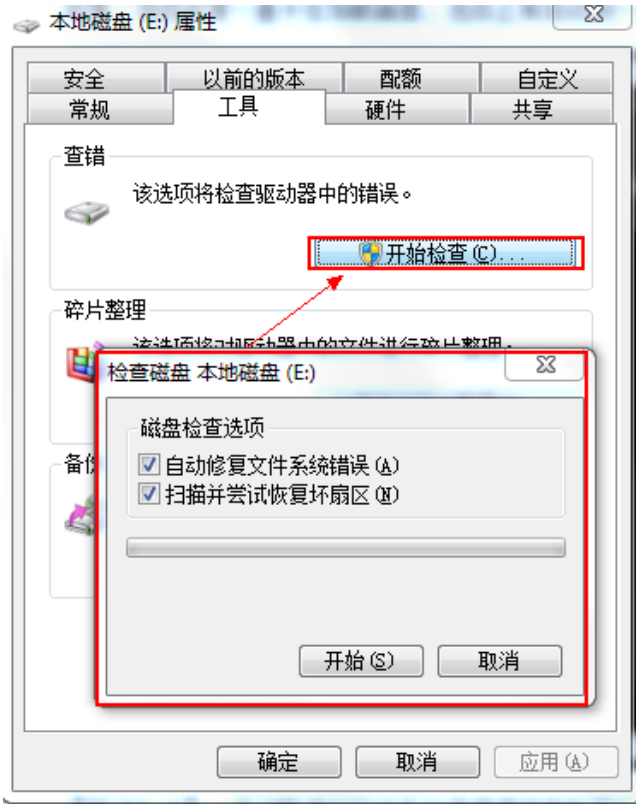
如果修复的对象为虚拟机，则可将问题虚拟机的系统盘镜像挂载到另外一台正常的 Windows 虚拟机上，使用 windows 自带的磁盘检查工具进行检查修复磁盘错误。通过[修改虚拟机/磁盘]页面执行“删除硬件”操作，将问题虚拟机的系统盘删除。



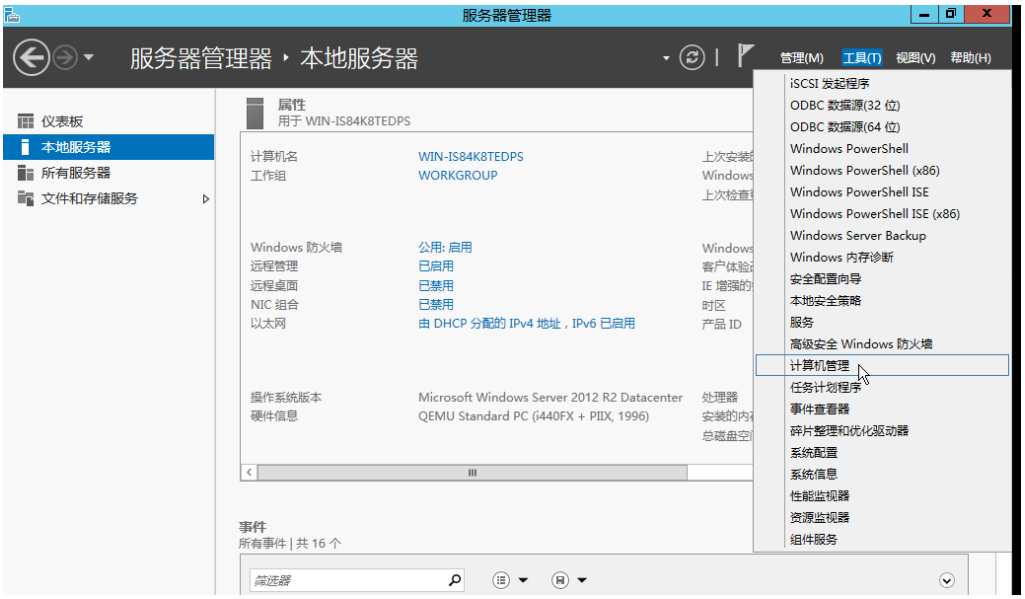
(2) 在正常的虚拟机上，通过添加硬盘的方式，添加问题虚拟机系统盘。



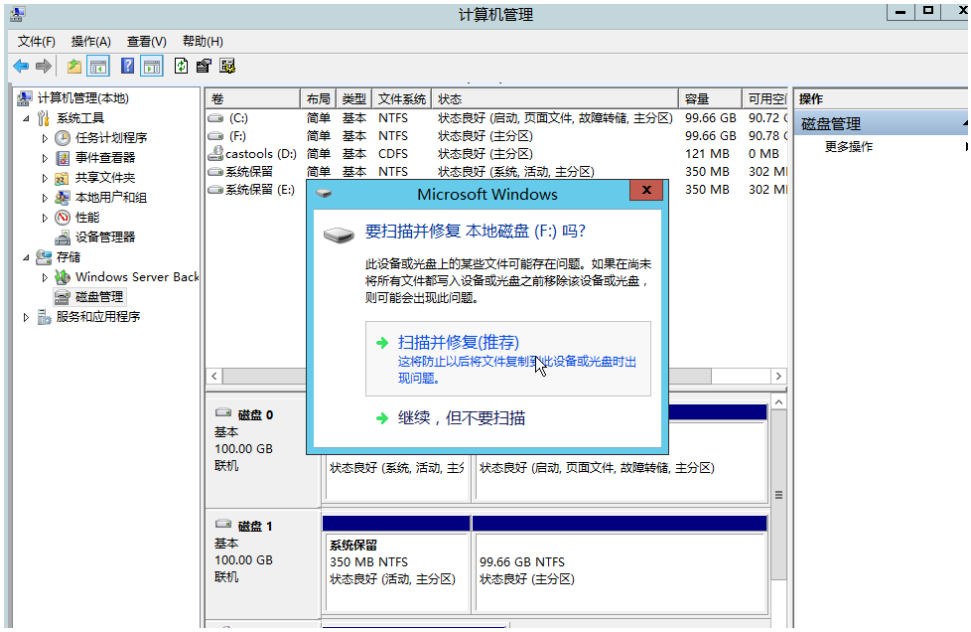
(3) 选择问题虚拟机镜像。此时在正常系统内部即可看到问题虚拟机的系统盘。



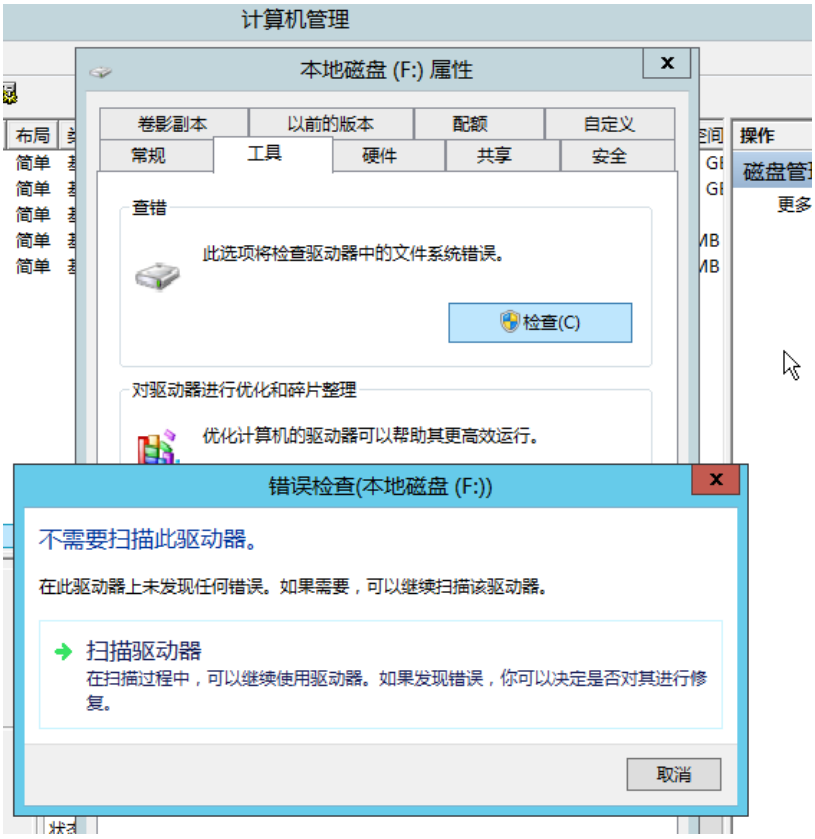
Windows 2012 是类似的界面，需要选择计算机管理部分，选择磁盘并查看属性，并进行错误检查。



(4) 将磁盘联机后，提示驱动器错误，点击蓝色报错区域。



也可选择 2 个分区的属性，进行扫描和修复处理。

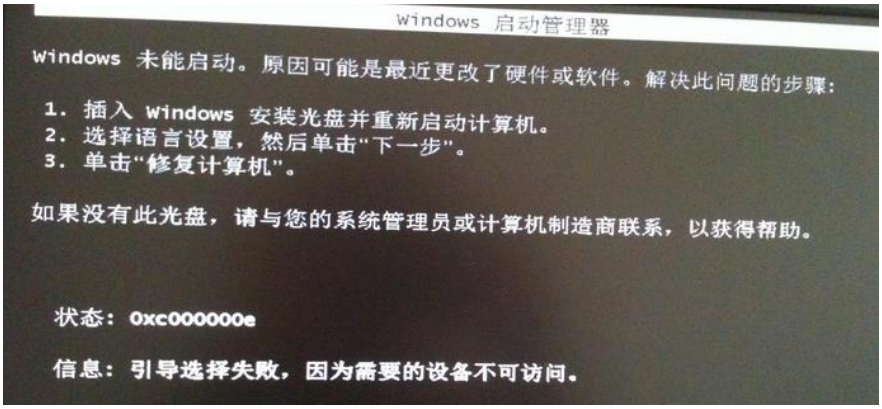


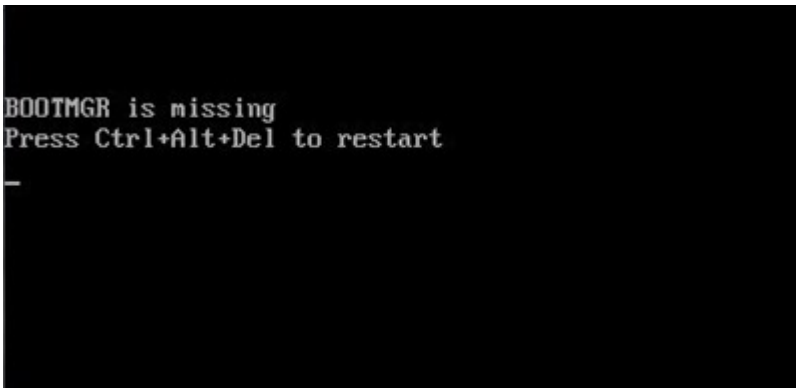
注意

- 操作过程和镜像文件，请采用正版系统的 ISO 进行。
- 在虚拟机化环境，对于 qcow2 格式的文件，多台虚拟机无法共同挂载，需要先解除一台的挂载关系，再挂给另外一台虚拟机，进行修复；而 RAW 格式/预分配置零的格式镜像文件/裸块方式的镜像则可以同时挂给另外的虚拟机。

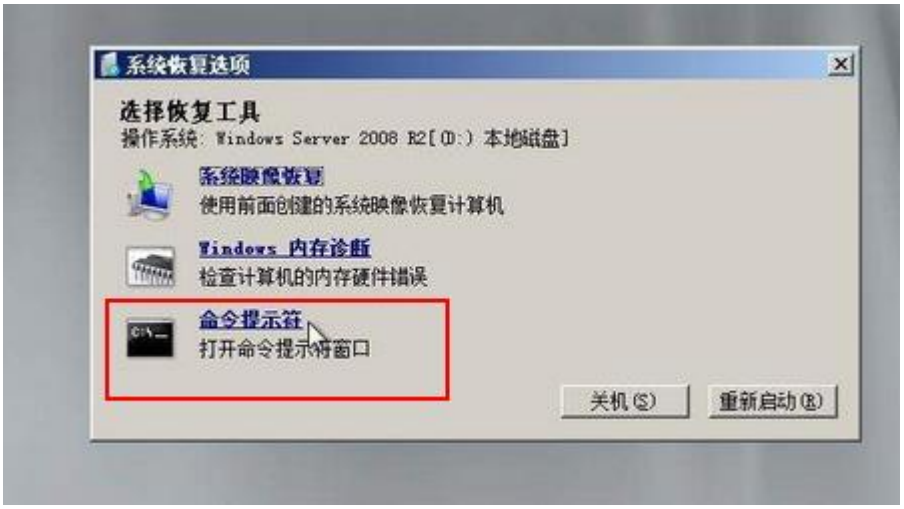
---

(5) 修复后的系统如果还报错误，则需要挂载 ISO 进行修复。将修复后的磁盘加回问题虚拟机，启动的时候会出现黑屏提示引导失败，或者另外一种情况是提示 bootmgr 丢失。

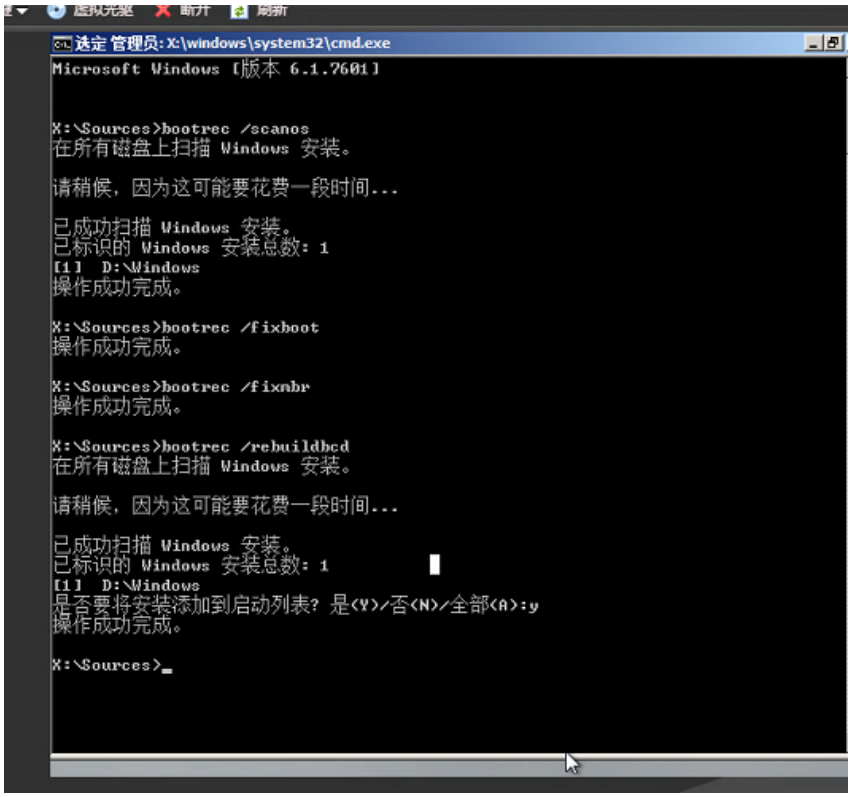




(6) 这时需要将系统光盘挂载到光驱进行修复 bootmgr，需要修改系统的引导顺序为从光盘引导。Windows2008 安装->修复计算机->选择命令提示符窗口。



(7) 输入如下命令修复 bootmgr，修复完后即可正常引导启动。



注意

- 虚拟化场景选择 IDE 磁盘，同时挂载合适版本的 ISO。
- 部分场景修复后启动还报其他错误，比如杀毒软件或应用程序的启动错误，可挂给一台好的 Windows 系统，进入出错所在目录下内部，将相关的软件或程序关闭或卸载掉（修改名称，使其不能启动）；再次尝试启动系统，可以根据具体的出错信息，进行相应的修改和调整。

7 常用命令

7.1 UIS常用命令

7.1.1 HA 相关命令

H3C UIS 使用了 H3C 自研 HA 功能，如下介绍该 HA 常用的命令。

如下命令中只有最后一条命令在 CVK 主机运行，其他命令均在 UIS Manager 主机运行。

1. 获取 HA 进程管理的集群列表

```
cha cluster-list
```

举例：

```
root@UIS-UISManager:~# cha cluster-list
------------------------------------------------------------
HA database info:
 Cluster list:
```

```
  cluster:1,    name:Cluster

HA memory info:
 Cluster list:
  cluster ID: 1
```

2. 查询指定的集群的状态信息：

可以查询指定集群包含了哪些主机和虚拟机信息。

```
cha cluster-status cluster-id
```

举例：

```
root@UIS-UISManager:~# cha cluster-status 1
------------------------------------------------------------
HA database info:
 Cluster 1 information:
  Is HA enabled: 1
  Cluster priority: 1
  2 nodes configured
  6 VM configured
 host and vm list:
  Host:UIS-CVK01, vm:windows2008
  Host:UIS-CVK02, vm:win2008
  Host:UIS-CVK02, vm:rhce-lab
  Host:UIS-CVK02, vm:Linux-RedHat5.9
  Host:UIS-CVK02, vm:fundation1
  Host:UIS-CVK02, vm:win7

HA memory info:
 Cluster 1, Least_host_number(MIN_HOST_NUM) is 1.
```

3. 查询集群中的主机列表：

可以查询指定集群包含了哪些主机和虚拟机信息。

```
cha node-list cluster-id
```

举例：

```
root@UIS-UISManager:~# cha node-list 1
------------------------------------------------------------
HA database info:
 In cluster 1, node list :
  host: UIS-CVK01, in cluster: 1, IP: 192.168.11.1
  host: UIS-CVK02, in cluster: 1, IP: 192.168.11.2

HA memory info:
 Cluster 1, Least_host_number(PermitNum) is 1. hosts list:
  host: UIS-CVK02  ID: 4
  host: UIS-CVK01  ID: 3
 Total host num in this cluster is: 2
```

4. 查询集群中某主机的信息：

```
cha node-status host-name
```

举例：

```
root@UIS-UISManager:~# cha node-status UIS-CVK01
------------------------------------------------------------
HA database info:
 Node UIS-CVK01 :
  in  cluster: 1
  ip address: 192.168.11.1
  VM count: 1

HA memory info:
 Host: UIS-CVK01, ID: 3, IP address: 192.168.11.1
  status: CONNECT
  heart beat num: 101
  storage total num: 1
  storage fail num: 0
  heartbeat fail num: 0
  recv packet: 1
  host model(maintain): 0
  time statmp: Fri Jan 30 15:34:04 2015
 Storage info:
  storage name:sharefile path:/vms/sharefile
  storage status:STORAGE_NORMAL
  time stamp:0
  update flag:0
  last send flag:0
  fail num:0
```

5. 打印某个主机内的虚拟机：

```
cha vm-list host-name
```

举例：

```
root@UIS-CVK03:~# cha vm-list UIS-CVK01
------------------------------------------------------------
HA database info:
 1 vms in host UIS-CVK01 :
  vm: windows2008   ID: 11   HA-managed: 1   Target-role: 1
```

6. 查询集群中的虚拟机信息：

```
cha vm-status vm-name
```

举例：

```
root@UIS-CVK03:~# cha vm-status windows2008
------------------------------------------------------------
HA database info:
 vm ID: 11  name: windows2008
  at node ID: 3
  target-role:  1
  is-managed: 1
```

```
  prority: 1
  storage name: sharefile
  storage psth: /vms/sharefile
```

7. 设置日志级别：

```
cha set-loglevel module level
```

参数：

cmd|UIS managerd：设置 cmd 进程或者 UIS Manager 进程的日志级别。

level：日志级别，debug|info|trace|warning|error|fatal。

举例：

```
root@UIS-UIS Manager:~# cha set-loglevel info
```

8. 设置 CVK 上日志级别：

```
cha -k set-loglevel level
```

参数：

level：日志级别，debug|info|trace|warning|error|fatal。

举例：

```
root@UIS-CVK01:/vms/sharefile# cha -k set-loglevel debug
Set cvk log level success.
root@UIS-CVK01:/vms/sharefile#
```

7.1.2 vSwitch 相关命令

H3C UIS 包含了 vSwitch 模块，如下介绍 vSwitch 模块的基本命令。

1. 查看 OVS 内部版本号

```
root@hz-cvknode2:~# ovs-vsctl -V
ovs-vsctl (Open vSwitch) 2.9.1
DB Schema 7.15.1
```

2. 查看虚拟交换机相关进程的状态

在 CVK 主机中执行“ps aux | grep ovs”命令，其中“ovs_workq”进程为 ovs 内核线程，“ovsdb-server”和“ovs-vswitchd”的进程各有两个，其一为监视进程，另一位业务进程。

```
root@UIS-CVK01:~# ps aux | grep ovs
root       2207  0.0  0.0      0     0 ?        S    Dec07   0:00 [ovs_workq]
root       3411  0.0  0.0  23228   772 ?        Ss   Dec07   6:44 ovsdb-server: monitoring
pid 3412 (healthy)
root       3412  0.0  0.0  23888  2656 ?        S    Dec07   6:15 /usr/sbin/ovsdb-server
/etc/openvswitch/conf.db --verbose=ANY:console:emer --verbose=ANY:syslog:err
--log-file=/var/log/openvswitch/ovsdb-server.log --detach --no-chdir --pidfile --monitor
--remote punix:/var/run/openvswitch/db.sock --remote
db:Open_vSwitch,Open_vSwitch,manager_options --remote ptcp:6632
--private-key=db:Open_vSwitch,SSL,private_key
--certificate=db:Open_vSwitch,SSL,certificate
--bootstrap-ca-cert=db:Open_vSwitch,SSL,ca_cert
root       3421  0.0  0.0  23972   804 ?        Ss   Dec07   7:23 ovs-vswitchd: monitoring
pid 3422 (healthy)
root       3422  0.4  0.0 1721128 9364 ?        Sl   Dec07  55:24 /usr/sbin/ovs-vswitchd
--verbose=ANY:console:emer --verbose=ANY:syslog:err
```

```
--log-file=/var/log/openvswitch/ovs-vswitchd.log --detach --no-chdir --pidfile --monitor
unix:/var/run/openvswitch/db.sock
root      23503  0.0  0.0   8112   936 pts/10   S+   10:43   0:00 grep --color=auto ovs
```

3. 重启虚拟交换机

```
root@UIS-CVK01:~# service openvswitch-switch restart
```

4. 增加虚拟交换机

```
root@UIS-CVK01:~# ovs-vsctl add-br vswitch-app
```

虚拟交换机增加完成后，在 UIS 页面重新连接主机后，可以查看到增加的虚拟交换机信息。

5. 删除虚拟交换机

```
root@UIS-CVK01:~# ovs-vsctl del-br vswitch-app
```

后台手工删除虚拟交换机后，UIS 前台无法同步。

6. 虚拟交换机添加端口

```
root@UIS-CVK01:~# ovs-vsctl add-port vswitch-app eth2
```

7. 虚拟交换机删除端口

```
root@UIS-CVK01:~# ovs-vsctl del-port vswitch-app eth2
```

后台手工删除虚拟交换机的端口后，UIS 前台无法同步。

8. 查看虚拟交换机和端口信息

其中 vswitch0 为内部口（又称 Local 口），eth0 为物理口，vnet0 为虚拟机端口。

```
root@UIS-CVK01:~# ovs-vsctl show
ba390c40-8826-4a7a-8e17-f8834dab6eb3
    Bridge "vswitch0"
        Port "eth0"
            Interface "eth0"
        Port "vswitch0"
            Interface "vswitch0"
                type: internal
        Port "vnet0"
            Interface "vnet0"
root@UIS-CVK01:~#
```

9. 查看虚拟交换机配置信息

```
root@UIS-CVK01:~# ovs-vsctl list br vswitch0
_uuid               : 3500114d-5619-460e-ada7-d1b97f63c93c
br_mode             : 【0】
controller          : 【】
datapath_id         : "0000ac162d88c35c"
datapath_type       : ""
drop_unknown_uniUISt: 【】
external_ids        : {}
fail_mode           : 【】
firewall_port       : 【】
flood_vlans         : 【】
flow_tables         : {}
ipfix               : 【】
mirrors             : 【】
name                : "vswitch0"
netflow             : 【】
other_config        : {}
ports        :【16a48463-f90b-42fe-9a12-ceacfd256235, 5495812e-29e0-4364-a89f-b54ea52dd344,
dec98186-2c83-447d-9215-28f99750a410】
protocols           : 【】
sflow               : 【】
status              : {}
stp_enable          : false
```

10. 查看端口配置信息

```
root@UIS-CVK01:~# ovs-vsctl list port vnet0
_uuid               : bc0b1e57-2d72-4fae-97b4-0bbca5d17ba1
TOS                 : routine
bond_downdelay      : 0
bond_fake_iface     : false
bond_mode           : []
bond_updelay        : 0
dynamic_acl_enable  : false
external_ids        : {}
fake_bridge         : false
interfaces          : [5495133f-7e81-4047-a0bd-734fae81f6f3]
lacp                : []
lan_acl_list        : []
lan_addr            : []
mac                 : []
name                : "vnet0"
other_config        : {}
qbg_mode            : [4]
qos                 : []
statistics          : {}
status              : {}
tag                 : [4]
tcp_syn_forbid      : false
```

```
trunks              : []
vlan_mode           : []
vm_ip               : []
vm_mac              : "0cda411dad80"
wan_acl_list        : []
wan_addr            : []
```

11. 查看端口用户态和内核态端口号

```
root@UIS-CVK01:~# ovs-appctl dpif/show
system@ovs-system: hit:10133796 missed:181938
        flows: cur: 11, avg: 12, max: 23, life span: 79639399ms
        hourly avg: add rate: 26.506/min, del rate: 26.462/min
        daily avg: add rate: 24.205/min, del rate: 24.210/min
        overall avg: add rate: 24.356/min, del rate: 24.354/min
        vswitch0: hit:6478229 missed:39021
                eth0 1/5: (system)
                vnet1 2/8: (system)
                vswitch0 65534/6: (internal)
```

以 eth0 端口为例，2 为其用户态端口号，即 openflow 端口号；5 为其内核态端口号，用户内核报文转发之用。

12. 查看虚拟交换机 MAC 信息

```
root@UIS-CVK01:~# ovs-appctl fdb/show vswitch0
 port  VLAN  MAC                Age
    1     0  00:0f:e2:5a:6a:20  134
    2     0  0c:da:41:1d:3d:18   95
    1     0  ac:16:2d:6f:3f:4a    6
    1     0  a0:d3:c1:f0:a6:ca    6
    1     0  c4:ca:d9:d4:c2:ff    2
    4     0  0c:da:41:1d:6d:94    2
LOCAL     0  2c:76:8a:5d:df:a2    2
    3     0  0c:da:41:1d:80:03    0
```

13. 查看虚拟交换机端口绑定信息

```
root@UIS-CVK02:~# ovs-appctl bond/show
---- vswitch-bond_bond ----
bond_mode: active-backup
bond-hash-basis: 0
updelay: 0 ms
downdelay: 0 ms
lacp_status: off

slave eth2: enabled
        active slave
        may_enable: true

slave eth3: disabled
        may_enable: false
```

14. 查看用户流表

```
root@UIS-CVK01:~# ovs-ofctl dump-flows vswitch0
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=752218.541s, table=0, n_packets=15106363, n_bytes=3556156038,
idle_age=0, hard_age=65534, priority=0 actions=NORMAL
```

15. 查看指定虚拟交换机的内核流表

```
root@UIS-CVK01:~# ovs-appctl dpif/dump-flows vswitch0
skb_priority(0),in_port(5),eth(src=74:25:8a:36:d8:9b,dst=ff:ff:ff:ff:ff:ff),eth_type(0x0
806),arp(sip=10.88.8.1/255.255.255.255,tip=10.88.8.206/255.255.255.255,op=1/0xff,sha=74:
25:8a:36:d8:9b/00:00:00:00:00:00,tha=00:00:00:00:00:00/00:00:00:00:00:00), packets:2,
bytes:120, used:3.018s, actions:6
skb_priority(0),in_port(5),eth(src=38:63:bb:b7:ed:6c,dst=01:00:5e:00:00:fc),eth_type(0x0
800),ipv4(src=10.88.8.140/0.0.0.0,dst=224.0.0.252/0.0.0.0,proto=17/0,tos=0/0,ttl=1/0,fra
g=no/0xff), packets:1, bytes:66, used:1.139s, actions:6
skb_priority(0),in_port(5),eth(src=c4:34:6b:6c:ef:a8,dst=ff:ff:ff:ff:ff:ff),eth_type(0x0
800),ipv4(src=10.88.8.200/0.0.0.0,dst=10.88.9.255/0.0.0.0,proto=17/0,tos=0/0,ttl=128/0,f
rag=no/0xff), packets:17, bytes:1564, used:3.370s, actions:6
skb_priority(0),in_port(5),eth(src=14:58:d0:b7:24:07,dst=ff:ff:ff:ff:ff:ff),eth_type(0x0
800),ipv4(src=10.88.8.229/0.0.0.0,dst=10.88.9.255/0.0.0.0,proto=17/0,tos=0/0,ttl=64/0,fr
ag=no/0xff), packets:6, bytes:692, used:0.771s, actions:6
skb_priority(0),in_port(5),eth(src=14:58:d0:b7:53:f6,dst=01:00:5e:7f:ff:fa),eth_type(0x0
800),ipv4(src=10.88.8.146/0.0.0.0,dst=239.255.255.250/0.0.0.0,proto=17/0,tos=0/0,ttl=1/0
,frag=no/0xff), packets:1, bytes:175, used:0.739s, actions:6
```

16. 查看所有内核流表

```
root@UIS-CVK01:~# ovs-dpctl dump-flows
skb_priority(0),in_port(4),eth(src=c4:34:6b:6c:f5:ab,dst=ff:ff:ff:ff:ff:ff),eth_type(0x0
800),ipv4(src=10.88.8.159/0.0.0.0,dst=10.88.9.255/0.0.0.0,proto=17/0,tos=0/0,ttl=128/0,f
rag=no/0xff), packets:25, bytes:2300, used:0.080s, actions:3
skb_priority(0),in_port(5),eth(src=14:58:d0:b7:53:f6,dst=33:33:00:01:00:02),eth_type(0x8
6dd),ipv6(src=fe80::288d:70d6:36ce:60f3/::,dst=ff02::1:2/::,label=0/0,proto=17/0,tclass=
0/0,hlimit=1/0,frag=no/0xff), packets:0, bytes:0, used:never, actions:6
skb_priority(0),in_port(13),eth(src=0c:da:41:1d:80:03,dst=c4:ca:d9:d4:c2:ff),eth_type(0x
0800),ipv4(src=192.168.2.15/255.255.255.255,dst=192.168.2.121/0.0.0.0,proto=6/0,tos=0/0,
ttl=128/0,frag=no/0xff), packets:1, bytes:54, used:2.924s, actions:2
skb_priority(0),in_port(4),eth(src=c4:34:6b:68:9b:78,dst=33:33:00:00:00:02),eth_type(0x8
6dd),ipv6(src=fe80::85b7:25a0:d116:907a/::,dst=ff08::2/::,label=0/0,proto=17/0,tclass=0/
0,hlimit=128/0,frag=no/0xff), packets:0, bytes:0, used:never, actions:3
skb_priority(0),in_port(4),eth(src=5c:dd:70:b0:39:3d,dst=ff:ff:ff:ff:ff:ff),eth_type(0x0
806),arp(sip=192.168.11.149/255.255.255.255,tip=192.168.11.150/255.255.255.255,op=1/0xff
,sha=5c:dd:70:b0:39:3d/00:00:00:00:00:00,tha=00:00:00:00:00:00/00:00:00:00:00:00),
packets:1, bytes:60, used:0.264s, actions:3
```

17. 端口抓包

使用 Linux 的 tcpdump 命令来对虚拟交换机对应的端口进行抓包，如下所示。tcpdump 命令的详细参数记录在[Linux 常用命令/网络相关命令]中。

```
tcpdump -i vnet1 -s 0 -w /tmp/test.pcap host 200.1.1.1 &
```

7.1.3 iSCSI 相关命令

H3C UIS 通过 iSCSI 技术来挂载 IP SAN 存储设备，当 iSCSI 共享文件系统出现问题时，可以通过 iSCSI 相关的命令进行排查。

1. 发现 iscsi 存储

```
iscsiadm -m discovery -t st -p ISCSI_IP
```

举例：

```
root@HZ-UIS01-CVK01:~# iscsiadm -m discovery -t st -p 192.168.1.248:3260
192.168.1.248:3260,1 iqn.1991-05.com.microsoft:c09599-cmh-target
root@HZ-UIS01-CVK01:~#
```

2. 查看 iscsi 发现记录

```
iscsiadm -m node
```

举例：

```
root@HZ-UIS01-CVK01:~# iscsiadm -m node
192.168.1.248:3260,1 iqn.1991-05.com.microsoft:c09599-cmh-target
```

3. 删除 iscsi 发现的记录

```
iscsiadm -m node -o delete -T LUN_NAME -p ISCSI_IP
```

举例：

```
# iscsiadm -m node -o delete -T iqn.1991-05.com.microsoft:c09599-cmh-target -p
192.168.1.248:3260
```

4. 登录 iscsi 存储

```
iscsiadm -m node -T LUN_NAME -p ISCSI_IP -l
```

举例：

```
root@HZ-UIS01-CVK01:~# iscsiadm -m node -T iqn.1991-05.com.microsoft:c09599-cmh-target
-p 192.168.1.248:3260 -l
Logging in to 【iface: default, target: iqn.1991-05.com.microsoft:c09599-cmh-target, portal:
192.168.1.248,3260】
Login to 【iface: default, target: iqn.1991-05.com.microsoft:c09599-cmh-target, portal:
192.168.1.248,3260】: successful
```

5. 退出 iscsi 存储

```
iscsiadm -m node -T LUN_NAME -p ISCSI_IP -u
```

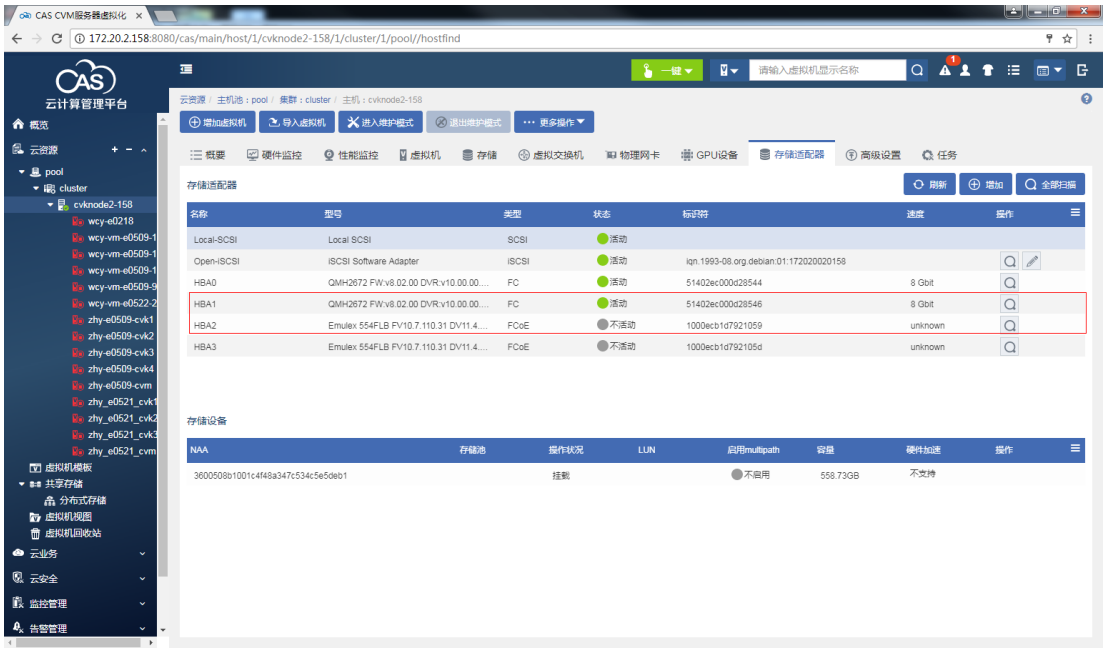
举例：

```
root@HZ-UIS01-CVK01:~# iscsiadm -m node -T iqn.1991-05.com.microsoft:c09599-cmh-target
-p 192.168.1.248:3260 -u
Logging out of session【sid: 4, target: iqn.1991-05.com.microsoft:c09599-cmh-target, portal:
192.168.1.248,3260】
Logout of 【sid: 4, target: iqn.1991-05.com.microsoft:c09599-cmh-target, portal:
192.168.1.248,3260】: successful
```

7.1.4 FC 挂载

1. 查询 HBA 卡

(1) 方法一：借助 CVM 查看，进入主机云计算管理平台，点击存储适配器，即可看到主机配置的 HBA 卡信息和状态，活动表示可以进行存储访问。如下图所示。



(2) 方法二：通过后台查询 HBA 卡驱动信息是否正确加载，如果 HBA 加载了正确的驱动，则在 /sys/class/fc_host/host*目录下有大量的 HBA 信息，host0 信息如下图所示：

```
[root@cvknode2-158 /]#ls /sys/class/fc_host/
host0  host2  host3  host4
 [root@cvknode2-158 /]#ls /sys/class/fc_host/host0
device       issue_lip       npiv_vports_inuse  port_state  speed       supported_classes
system_hostname  vport_create
dev_loss_tmo  max_npiv_vports  port_id           port_type   statistics  supported_speeds
tgtid_bind_type  vport_delete
fabric_name  node_name      port_name       power      subsystem  symbolic_name     uevent
```

2. 连接 fc 存储

使用如下命令连接 FC 存储。

```
echo hba_channel  target_id target_lun > /sys/class/scsi_host/host*/scan
```

其中 hba_channel 表示 HBA 卡通道，target_id 表示设备 id，target_lun 表示设备 lun，可通过查询 /sys/class/fc_transport/获取。

```
[root@cvknode2-158 /]#ls /sys/class/fc_transport/
target0:0:0
[root@cvknode2-158 /]# echo 0 0 0 > /sys/class/scsi_host/host0/scan
```

3. 断开 fc 存储

使用如下命令断开 FC 存储的连接。

```
echo 1 > /sys/block/sdX/device/delete
```

其中 sdX 表示需要断开 fc 设备所对应的 sd 设备，可借助 ll 命令查询。

```
[root@cvknode2-158 /]# ll /dev/disk/by-path
lrwxrwxrwx 1 root root   9 Oct 12 09:48 pci-0000:05:00.0-fc-0x21020002ac01e2d7-lun-0
-> ../../sdb
[root@cvknode2-158 /]# echo 1 > /sys/block/sdb/device/delete
```

7.1.5 Tomcat 服务命令

H3C UIS 的 WEB 页面使用了 Tomcat 技术，当 UIS 的 WEB 页面出现异常时，可以重启 Tomcat 服务。

查看 Tomcat 服务状态命令。

```
root@HZ-UIS01-CVK01:~# service tomcat8 status
 * Tomcat servlet engine is running with pid 3362
```

停止 Tomcat 服务状态命令。

```
root@HZ-UIS01-CVK01:~# service tomcat8 stop
 * Stopping Tomcat servlet engine tomcat8
   ...done.
```

启动 Tomcat 服务状态命令。

```
root@HZ-UIS01-CVK01:~# service tomcat8 start
 * Starting Tomcat servlet engine tomcat8
   ...done.
```

重启 Tomcat 服务状态命令。

```
root@ HZ-UIS01-CVK01:~# service tomcat8 restart
 * Stopping Tomcat servlet engine tomcat8
   ...done.
 * Starting Tomcat servlet engine tomcat8
   ...done.
root@ HZ-UIS01-CVK01:~#
```

7.1.6 MySQL 数据库服务命令

H3C UIS 的使用了 MySQL 数据技术，需要掌握 MySQL 数据库的基本操作方法。

查看 MySQL 服务状态命令。

```
root@HZ-UIS01-CVK01:~# service mysql status
mysql start/running, process 3039
```

停止 MySQL 服务命令。

```
root@HZ-UIS01-CVK01:~#
root@HZ-UIS01-CVK01:~# service mysql stop
mysql stop/waiting
```

启动 MySQL 服务命令。

```
root@HZ-UIS01-CVK01:~# service mysql start
mysql start/running, process 4821
```

7.1.7 virsh 相关命令

通过 virsh 相关的命令，可以从 CVK 后台查看该 CVK 主机下包含了哪些虚拟机，以及虚拟机的状态，并且可以启动或者关闭虚拟机。

1. CVK 主机后台查看虚拟机

在 CVK 主机后台执行“virsh list --all”命令查看该 CVK 主机下虚拟机状态。

```
root@UIS-CVK01:/vms# virsh list --all
 Id    Name                           State
----------------------------------------------------
 4     windows2008                    running
 -     Linux-RedHat5.9                shut off
```

2. CVK 主机后台启动虚拟机

在 CVK 主机后台执行“virsh start 虚拟机名称”命令启动虚拟机。

```
root@UIS-CVK01:/vms# virsh start Linux-RedHat5.9
Domain Linux-RedHat5.9 started
root@UIS-CVK01:/vms#
```

3. CVK 主机后台关闭虚拟机

在 CVK 主机后台执行“virsh shutdown 虚拟机名称”命令关闭虚拟机。

```
root@UIS-CVK01:/vms# virsh shutdown Linux-RedHat5.9
Domain Linux-RedHat5.9 is being shutdown
```

7.1.8 casserver 服务启动命令

caserver 管理相关信息的收集，比如磁盘利用率，告警信息等。当 caserver 出现问题的时候需要重启服务，命令如下。

```
service casserver restart
```

7.1.9 qemu 相关命令

通过 qemu 相关命令可以查看虚拟机对应的镜像文件信息，并且可以对镜像文件格式进行转换操作。

1. 查看虚拟机的镜像文件信息

在 UIS 的 WEB 页面，可以查看到虚拟机对应的镜像文件路径，如下图所示，虚拟机“A_048”对应的镜像文件路径为“/vms/defaultShareFileSystem0/A_048”。

通过 qemu-info 命令，可以查看镜像文件的基本信息，比如文件格式，文件总大小以及当前使用的大小。如果是三级镜像文件还会显示二级镜像文件名称信息。

```
root@ZJ-UIS-001:~# qemu-img info /vms/defaultShareFileSystem0/A_048
image: /vms/defaultShareFileSystem0/A_048
file format: qcow2
virtual size: 30G (32212254720 bytes)
disk size: 1.3G
cluster_size: 262144
backing file: /vms/defaultShareFileSystem0/A_048_base_1
backing file format: qcow2
Format specific information:
    compat: 0.10
    refcount bits: 16
```

继续查看二级镜像文件信息，可以查看到一级镜像文件（即基础镜像文件）的信息。

```
root@ZJ-UIS-001:~# qemu-img info /vms/defaultShareFileSystem0/A_048_base_1
image: /vms/defaultShareFileSystem0/A_048_base_1
file format: qcow2
virtual size: 30G (32212254720 bytes)
disk size: 1.0M
cluster_size: 262144
backing file:
/vms/defaultShareFileSystem0/fio-cent-autorun_UIS-e0602fio-cent-autorun_UIS-e0602
backing file format: qcow2
Format specific information:
    compat: 0.10
    refcount bits: 16
```

继续查看一级镜像文件（即基础镜像文件）信息，不再包含下一级的镜像文件。

```
root@ZJ-UIS-001:~# qemu-img info
/vms/defaultShareFileSystem0/fio-cent-autorun_UIS-e0602fio-cent-autorun_UIS-e0602
image: /vms/defaultShareFileSystem0/fio-cent-autorun_UIS-e0602fio-cent-autorun_UIS-e0602
file format: qcow2
virtual size: 30G (32212254720 bytes)
```

```
disk size: 5.5G
cluster_size: 262144
Format specific information:
    compat: 1.1
    lazy refcounts: false
    refcount bits: 16
    corrupt: false
```

2. 合并镜像文件

当虚拟机是多级镜像文件时，可以通过 qemu-img 命令对多级镜像文件进行合并。

```
root@UIS-CVK01:/vms/sharefile# qemu-img convert -O qcow2 -f qcow2 windows2008 windows2008-test
root@ZJ-UIS-001:/vms/defaultShareFileSystem0# qemu-img convert -O qcow2 -f qcow2 A_048 A048-test
```

合并完成后输出的镜像文件不是多级镜像文件。

```
root@ZJ-UIS-001:~# qemu-img info  /vms/defaultShareFileSystem0/A048-test
image: /vms/defaultShareFileSystem0/A048-test
file format: qcow2
virtual size: 30G (32212254720 bytes)
disk size: 1.4G
cluster_size: 262144
Format specific information:
    compat: 1.1
    lazy refcounts: false
    refcount bits: 16
    corrupt: false
```

7.1.10 ONEStor 相关命令

ONEStor 集群常用的运维命令包括查询集群健康状态、查看 mon、OSD、PG 状态信息等操作。

- mon（Monitor）：集群的监控节点；
- OSD：集群存储节点对应的各个物理硬盘；
- PG：概览页面显示的虚节点，PG 位于存储池内，每增加一个存储池，集群都会对应新增一定数量的 PG。

1. 集群健康状态查询

(1) ceph health detail

获取集群的详细健康状态信息，命令会列出所有不是 active 和 clean 状态的 PG，包括所有处于 unclean、inconsistent、degraded 状态的 PG，如下图集群状态是健康会输出“HEALTH_OK”。

```
root@node110:~# ceph health detail
HEALTH_OK
```

集群非健康状态如下：

HEALTH_WARN 表示集群处于“警告”状态，下图为 1024 个 PG 处于 degraded 降级状态，1024 个 PG 处于 unclean 状态，集群中 33.333%的对象被降级，1/3 的 OSD 处于 down 状态，down 的 OSD 上所对应的 PG 处于 degraded 状态。

三节点集群，1/3 的 OSD down，造成这种现象的可能原因：

- 单节点网络不通，确认集群业务网、存储网是否能持续 ping 通；
- 单节点故障，ceph osd tree 查看 down 的 OSD 处于哪个节点，查看该节点硬件、操作系统是否正常。

```
HEALTH_WARN 1024 pgs degraded; 1024 pgs stuck unclean; 1024 pgs undersized; recovery 128003/384009 objects degraded (33.333%); 1/
3 in osds are down
pg 1.17c is stuck unclean for 24133.776596, current state active+undersized+degraded, last acting [2,1]
pg 1.17d is stuck unclean for 985.992538, current state active+undersized+degraded, last acting [1,2]
pg 1.17a is stuck unclean for 24133.516198, current state active+undersized+degraded, last acting [1,2]
pg 1.17b is stuck unclean for 985.994030, current state active+undersized+degraded, last acting [1,2]
pg 1.178 is stuck unclean for 24133.626964, current state active+undersized+degraded, last acting [2,1]
pg 1.179 is stuck unclean for 24133.971844, current state active+undersized+degraded, last acting [2,1]
pg 1.176 is stuck unclean for 985.992694, current state active+undersized+degraded, last acting [1,2]
pg 1.177 is stuck unclean for 986.045428, current state active+undersized+degraded, last acting [2,1]
pg 1.174 is stuck unclean for 985.993350, current state active+undersized+degraded, last acting [1,2]
pg 1.175 is stuck unclean for 985.993208, current state active+undersized+degraded, last acting [1,2]
pg 1.172 is stuck unclean for 24133.948626, current state active+undersized+degraded, last acting [2,1]
pg 1.173 is stuck unclean for 24134.363868, current state active+undersized+degraded, last acting [2,1]
pg 1.170 is stuck unclean for 985.992134, current state active+undersized+degraded, last acting [1,2]
pg 1.171 is stuck unclean for 985.994297, current state active+undersized+degraded, last acting [1,2]
pg 1.16e is stuck unclean for 985.995314, current state active+undersized+degraded, last acting [1,2]
pg 1.16f is stuck unclean for 24133.852982, current state active+undersized+degraded, last acting [1,2]
pg 1.16c is stuck unclean for 986.042849, current state active+undersized+degraded, last acting [2,1]
pg 1.16d is stuck unclean for 986.044391, current state active+undersized+degraded, last acting [2,1]
pg 1.16a is stuck unclean for 985.996013, current state active+undersized+degraded, last acting [1,2]
pg 1.16b is stuck unclean for 24133.543649, current state active+undersized+degraded, last acting [2,1]
```

(2) ceph -s

ceph 集群管理中查看集群状态是使用频率最高的操作，使用 ceph -s 命令查看集群状态；集群状态信息如下：

```
root@node117:~# ceph -s
    cluster 7bc9f9d3-6a84-455f-bdf0-2e05f7c1cb90
     health HEALTH_OK
     monmap e1: 3 mons at {node117=192.168.101.117:6789/0,node118=192.168.101.118:6789/0,n
ode119=192.168.101.119:6789/0}
            election epoch 6, quorum 0,1,2 node117,node118,node119
     osdmap e650: 18 osds: 18 up, 18 in
      pgmap v38166: 1024 pgs, 1 pools, 102399 MB data, 25603 objects
            202 GB used, 9672 GB / 9874 GB avail
                1024 active+clean
```

ceph -s 输出集群状态信息，含义如下：

- health：health HEALTH_OK 表示集群健康，health HEALH_WARN 表示集群有告警信息，还有 HEALTH_ERR 状态表示集群发生数据不一致等情况下的严重错误状态，根据输出信息确定集群异常原因，通常情况下 PG 异常，OSD 异常，集群时间不一致等异常都会在 health 处有提示；
- monmap：查看集群 monitor 数量及所在节点。上图示例查看到，集群有 3 个 monitor，分别位于 node117，node118，node119 上，显示的第一个为主 monitor；
- osdmap：查看 OSD 总数，处于 up(OSD 在运行)状态和 in(OSD 在集群中)状态的 OSD 数量。上图示例查看到；集群共有 18 个 OSD，处于 up 且 in 状态，此时 OSD 全部正常；
- pgmap：查看集群 PG 数量、存储池数量、一份数据副本所占的空间大小、对象的总数量，还显示了集群使用的信息，包括已用容量、可用容量和总容量。最后还显示了 PG 的状态信息；例如，图中集群 PG 数量 1024 个，1 个存储池，集群使用容量 202G，object 数量 25603，可用容量 9672G；

常见错误提示：

- too many PGs per OSD：如下显示为每个 OSD 对应的 PG 数量过多，增加 OSD 个数或者减少存储池个数后告警提示会消失。

```
root@node110:~# ceph -s
    cluster 8f4a67ac-3477-4ce9-8bcf-3da44b30ff34
     health HEALTH_WARN
            too many PGs per OSD (1706 > max 1000)
     monmap e2: 2 mons at {node111=192.168.101.111:6789/0,node121=192.168.101.121:6789/0}
            election epoch 124, quorum 0,1 node111,node121
     osdmap e2196: 3 osds: 3 up, 3 in
      pgmap v34881: 2048 pgs, 2 pools, 499 GB data, 125 kobjects
            1003 GB used, 1758 GB / 2762 GB avail
                2048 active+clean
```

- clock skew detected：mon 节点之间存在时间不一致，需要向主 ntp server 同步，执行 ntpdate –u IP（IP 为主 ntp server 地址）同步时间。例如下图有 6 个 OSD 处于 down 状态，集群将这些 down 掉的 OSD 所对应的 PG 置于 degraded 状态。

```
root@node117:~# ceph -s
    cluster 7bc9f9d3-6a84-455f-bdf0-2e05f7c1cb90
     health HEALTH_WARN
            679 pgs degraded
            400 pgs stuck unclean
            679 pgs undersized
            recovery 16985/51206 objects degraded (33.170%)
            6/18 in osds are down
     monmap e1: 3 mons at {node117=192.168.101.117:6789/0,node118=192.168.101.118:6789/0,node119=192
.168.101.119:6789/0}
            election epoch 6, quorum 0,1,2 node117,node118,node119
     osdmap e655: 18 osds: 12 up, 18 in
      pgmap v38177: 1024 pgs, 1 pools, 102399 MB data, 25603 objects
            202 GB used, 9672 GB / 9874 GB avail
            16985/51206 objects degraded (33.170%)
                 679 active+undersized+degraded
                 345 active+clean
```

下图 ceph -s 查看集群有 PG 异常，1 个 mon down，osd 12 up，18 in，考虑 node118 节点是否故障/业务网状态是否正常；

```
root@node117:~# ceph -s
    cluster 7bc9f9d3-6a84-455f-bdf0-2e05f7c1cb90
     health HEALTH_WARN
            679 pgs degraded
            400 pgs stuck unclean
            679 pgs undersized
            recovery 16985/51206 objects degraded (33.170%)
            6/18 in osds are down
            1 mons down, quorum 0,2 node117,node119
     monmap e1: 3 mons at {node117=192.168.101.117:6789/0,node118=192.168.101.118:6789/0,node119=1
.168.101.119:6789/0}
            election epoch 8, quorum 0,2 node117,node119
     osdmap e655: 18 osds: 12 up, 18 in
      pgmap v38177: 1024 pgs, 1 pools, 102399 MB data, 25603 objects
            202 GB used, 9672 GB / 9874 GB avail
            16985/51206 objects degraded (33.170%)
                 679 active+undersized+degraded
                 345 active+clean
```

(3) ceph -w

ceph -w 用于监控集群的状态变化，命令会持续输出，使用 ctrl+c 结束命令。下图集群 PG 状态正常时 ceph -w 显示与 ceph -s 开头显示一致。

```
^Croot@node117:~# ceph -w
    cluster 7bc9f9d3-6a84-455f-bdf0-2e05f7c1cb90
     health HEALTH_OK
     monmap e1: 3 mons at {node117=192.168.101.117:6789/0,node118=192.168.101.118:6789/0,node119=1
92.168.101.119:6789/0}
            election epoch 12, quorum 0,1,2 node117,node118,node119
     osdmap e665: 18 osds: 18 up, 18 in
      pgmap v38219: 1024 pgs, 1 pools, 102399 MB data, 25603 objects
            202 GB used, 9672 GB / 9874 GB avail
                1024 active+clean
```

下图集群异常，可以通过 osdmap、pgmap、mon、osd pgmap 的实时查看集群的变化。

```
^Croot@node110:~# ceph -w
    cluster 8f4a67ac-3477-4ce9-8bcf-3da44b30ff34
     health HEALTH_WARN
            too many PGs per OSD (1706 > max 1000)
     monmap e2: 2 mons at {node111=192.168.101.111:6789/0,node121=192.168.101.121:6789/0}
            election epoch 126, quorum 0,1 node111,node121
     osdmap e2196: 3 osds: 3 up, 3 in
      pgmap v34902: 2048 pgs, 2 pools, 499 GB data, 125 kobjects
            1003 GB used, 1758 GB / 2762 GB avail
                2048 active+clean

2017-05-08 04:02:20.693669 mon.0 [INF] pgmap v34902: 2048 pgs: 2048 active+clean; 499 GB data, 1003 GB used, 1758 GB / 27
62 GB avail
2017-05-08 04:03:56.064885 mon.1 [INF] mon.node121 calling new monitor election
2017-05-08 04:03:56.065354 mon.0 [INF] mon.node111 calling new monitor election
2017-05-08 04:03:56.198883 mon.0 [INF] mon.node111@0 won leader election with quorum 0,1
2017-05-08 04:03:56.222293 mon.0 [INF] HEALTH_WARN; too many PGs per OSD (1706 > max 1000)
2017-05-08 04:03:56.268199 mon.0 [INF] monmap e2: 2 mons at {node111=192.168.101.111:6789/0,node121=192.168.101.121:6789/
0}
2017-05-08 04:03:56.268264 mon.0 [INF] pgmap v34902: 2048 pgs: 2048 active+clean; 499 GB data, 1003 GB used, 1758 GB / 27
62 GB avail
2017-05-08 04:03:56.268332 mon.0 [INF] mdsmap e1: 0/0/0 up
2017-05-08 04:03:56.268423 mon.0 [INF] osdmap e2196: 3 osds: 3 up, 3 in
2017-05-08 04:03:59.383604 mon.0 [INF] pgmap v34903: 2048 pgs: 2048 active+clean; 499 GB data, 1003 GB used, 1758 GB / 27
62 GB avail
2017-05-08 04:04:02.601309 mon.0 [INF] osd.1 192.168.101.111:7300/2825281 failed (3 reports from 2 peers after 26.803379
>= grace 20.000000)
2017-05-08 04:04:02.744299 mon.0 [INF] osdmap e2197: 3 osds: 2 up, 3 in
2017-05-08 04:04:02.782116 mon.0 [INF] pgmap v34904: 2048 pgs: 681 stale+active+clean, 1367 active+clean; 499 GB data, 10
03 GB used, 1758 GB / 2762 GB avail
2017-05-08 04:04:03.767588 mon.0 [INF] osdmap e2198: 3 osds: 2 up, 3 in
2017-05-08 04:04:03.785119 mon.0 [INF] pgmap v34905: 2048 pgs: 681 stale+active+clean, 1367 active+clean; 499 GB data, 10
03 GB used, 1758 GB / 2762 GB avail
```

2. OSD 相关命令

(1) ceph osd tree

ceph osd tree 命令显示各个节点上的 OSD 以及其在 CRUSH map 中的位置，显示 OSD 权重(weight)，up/down，in/out 状态，对于维护一个大规模集群非常有帮助。OSD 正常状态如下：

```
root@node117:~# ceph osd tree
ID WEIGHT  TYPE NAME           UP/DOWN REWEIGHT PRIMARY-AFFINITY
-8       0 root maintain
-7       0 root ssd_root
-6       0     rack r0_ssd
-1 9.71988 root default
-5 9.71988     rack r0
-4 3.23996         host node117
14 0.53999             osd.14        up  1.00000          1.00000
17 0.53999             osd.17        up  1.00000          1.00000
 0 0.53999             osd.0         up  1.00000          1.00000
 1 0.53999             osd.1         up  1.00000          1.00000
 2 0.53999             osd.2         up  1.00000          1.00000
 3 0.53999             osd.3         up  1.00000          1.00000
-2 3.23996         host node119
12 0.53999             osd.12        up  1.00000          1.00000
15 0.53999             osd.15        up  1.00000          1.00000
 8 0.53999             osd.8         up  1.00000          1.00000
 9 0.53999             osd.9         up  1.00000          1.00000
10 0.53999             osd.10        up  1.00000          1.00000
11 0.53999             osd.11        up  1.00000          1.00000
-3 3.23996         host node118
13 0.53999             osd.13        up  1.00000          1.00000
16 0.53999             osd.16        up  1.00000          1.00000
 4 0.53999             osd.4         up  1.00000          1.00000
 5 0.53999             osd.5         up  1.00000          1.00000
 6 0.53999             osd.6         up  1.00000          1.00000
 7 0.53999             osd.7         up  1.00000          1.00000
```

例如如下以 osd.1 为例，osd.1 权重 0.89999，位于机架 rack 3 上，节点 host node111，处于 down 且 out 状态

```
root@node110:~# ceph osd tree
ID WEIGHT  TYPE NAME           UP/DOWN REWEIGHT PRIMARY-AFFINITY
-6       0 root maintain
-5 2.70000 root p1
-7 2.70000     rack 3
-4 0.89999         host node111
 1 0.89999             osd.1       down        0          1.00000
-3 0.89999         host node110
 2 0.89999             osd.2       down  1.00000          1.00000
-2 0.89999         host node121
 0 0.89999             osd.0         up  1.00000          1.00000
-1       0 root default
```

说明

OSD 处于 down 状态 5 分钟后会从 down in 被标记为 down out 状态，此时需考虑如下情况

- 单个 OSD down/out，考虑是否有硬盘故障。
- 整个节点 OSD down，考虑是否有节点异常、网络异常。

---

(2) ceph osd perf

有业务情况下延时（fs_apply_latency）稳定在 100ms 以内是认为是正常的，集群空闲时延时一般持续在 10ms 以内。

```
root@node117:~# ceph osd perf
osd fs_commit_latency(ms) fs_apply_latency(ms)
  0                   1                  0
  1                   0                  1
  2                   1                  0
  3                   1                  0
  4                   0                  0
  5                   0                  0
  6                   0                  0
  7                   0                  0
  8                   0                  1
  9                   0                  1
 10                   0                  1
 11                   0                  0
 12                   0                  0
 13                   0                  0
 14                   0                  1
 15                   0                  0
 16                   0                  0
 17                   0                  0
```

集群空闲状态下，延时持续大于 10ms 建议排查，业务量大的情况下，延时大于 100ms 建议排查，例如网络，硬件是否故障，集群是否健康。

(3) ceph osd df

通过 ceph osd df 命令查看集群磁盘使用情况。可查看集群 OSD 的使用数据，列出 OSD 的大小、已用容量、可用容量、使用率、集群有 OSD 使用率超过 85%前台会出现 near full 的告警，集群有 OSD 使用率超过 95%，集群不可读写。

例如下图中，集群包含 3 个 OSD，大小 920G，已用容量 501G，可用容量 419G，总容量 2762G，使用 1505G，剩余 1257G，使用率 54.48%；

```
root@node111:~# ceph osd df
ID WEIGHT  REWEIGHT SIZE  USE   AVAIL %USE  VAR
 1 0.89999  1.00000  920G  501G  419G 54.49 1.00
 2 0.89999  1.00000  920G  501G  419G 54.47 1.00
 0 0.89999  1.00000  920G  501G  419G 54.48 1.00
              TOTAL 2762G 1505G 1257G 54.48
MIN/MAX VAR: 1.00/1.00  STDDEV: 0.01
reweight_state : 0
```

3. 集群使用状况查询

(1) ceph df

统计集群及存储池的使用数据，显示集群的总容量，剩余容量和已使用的容量以及百分比，还显示存储池信息，比如存储池名称、ID、使用情况、每个存储池中的对象数量。

例如下图，集群剩余容量 1257G，已使用容量 1505G，使用百分比 54.48%，存储池 p1 下使用容量 499G，使用 54.29%，剩余可用 419G，对象数目 128003。

```
root@node111:~# ceph df
GLOBAL:
    SIZE      AVAIL     RAW USED     %RAW USED
    2762G     1257G        1505G         54.48
POOLS:
    NAME     ID     USED     %USED     MAX AVAIL     OBJECTS
    p1       1      499G     54.29          419G      128003
root@node111:~#
```

7.1.11 ONEStor 运维命令

1. iostat

iostat 工具查看进程 IO 请求下发的情况，系统处理 IO 请求的耗时，进而分析进程与操作系统的交互过程中 IO 是否存在瓶颈，单独执行 iostat，显示的结果为从系统开机到当前执行时刻的统计信息。iostat 输出如下图所示。

```
root@node118:~# iostat
Linux 3.19.0-32-generic (node118)       05/03/2017      _x86_64_        (24 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          0.19    0.00    0.08    0.02    0.00   99.72

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               1.13         4.35       134.49     290894    8985422
sdb               0.58         0.58        55.26      38484    3691939
sdc               0.59         0.62        55.24      41115    3690932
sdd               0.58         0.48        59.65      32365    3984963
sde               0.58         0.49        59.60      32961    3981868
sdf              86.31         1.17       396.89      78228   26516215
dm-0              1.94         4.23       134.49     282349    8985416
dm-1              0.00         0.02         0.00       1296          0
```

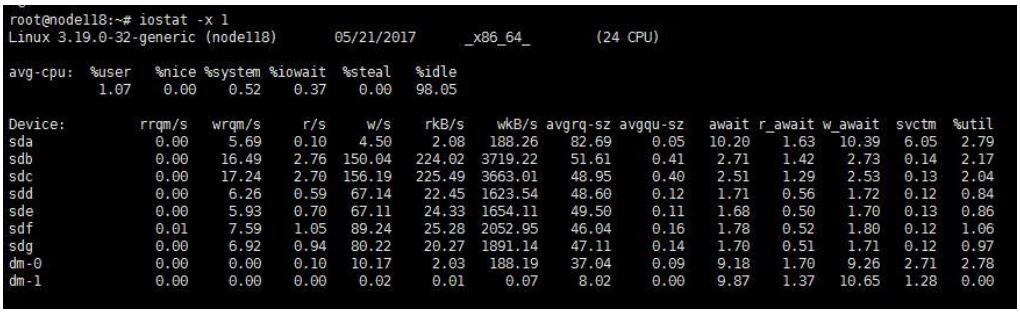
各项输出项目的含义如下：

- 最上面显示指示系统版本、主机名和日期。
- avg-cpu：总体 CPU 使用情况统计信息，对于多核 CPU，这里为所有 CPU 的平均值。
- Device：各磁盘设备的 IO 统计信息。
- 用户输出 CPU 和磁盘 IO 系统的负载统计信息。

对于 CPU 统计信息一行，我们主要看 iowait 的值，它指示 CPU 用于等待 IO 请求完成的时间。Device 列以 sdX 形式显示的设备名称。

项目	含义
tps	每秒进程下发的IO读、写请求数量
kB_read/s	每秒读扇区数量（一扇区为512bytes）
kB_wrtn/s	每秒写扇区数量
kB_read	取样时间间隔内读扇区总数量
kB_wrtn	取样时间间隔内写扇区总数量

iostat -x 1 实时显示详细的 io 设备统计信息，在分析 IO 瓶颈时，一般都会开启-x 选项。



```
root@node118:~# iostat -x 1
Linux 3.19.0-32-generic (node118)       05/21/2017      _x86_64_        (24 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          1.07    0.00    0.52    0.37    0.00   98.05

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sda               0.00     5.69    0.10    4.50     2.08   188.26    82.69     0.05   10.20    1.63   10.39   6.05   2.79
sdb               0.00    16.49    2.76  150.04   224.02  3719.22    51.61     0.41    2.71    1.42    2.73   0.14   2.17
sdc               0.00    17.24    2.70  156.19   225.49  3663.01    48.95     0.40    2.51    1.29    2.53   0.13   2.04
sdd               0.00     6.26    0.59   67.14    22.45  1623.54    48.60     0.12    1.71    0.56    1.72   0.12   0.84
sde               0.00     5.93    0.70   67.11    24.33  1654.11    49.50     0.11    1.68    0.50    1.70   0.13   0.86
sdf               0.01     7.59    1.05   89.24    25.28  2052.95    46.04     0.16    1.78    0.52    1.80   0.12   1.06
sdg               0.00     6.92    0.94   80.22    20.27  1891.14    47.11     0.14    1.70    0.51    1.71   0.12   0.97
dm-0              0.00     0.00    0.10   10.17     2.03   188.19    37.04     0.09    9.18    1.70    9.26   2.71   2.78
dm-1              0.00     0.00    0.00    0.02     0.01     0.07     8.02     0.00    9.87    1.37   10.65   1.28   0.00
```

iostat -x 1 实时显示节点硬盘使用情况，单个硬盘%util 比例偏高/接近 100%，可以考虑瓶颈在单块硬盘，集群整体硬盘%util 比例 80%以上或是接近 100%是说明集群硬盘 IO 使用率已经达到极限，瓶颈很可能在硬盘，可以考虑添加硬盘或者减少用户业务；

各项输出项目的含义如下：

项目	含义
rrqm/s	每秒对该设备的读请求被合并次数，文件系统会对读取同块(block)的请求进行合并
wrqm/s	每秒对该设备的写请求被合并次数
r/s	每秒完成的读次数
w/s	每秒完成的写次数
rkB/s	每秒读数据量(kB为单位)
wkB/s	每秒写数据量(kB为单位)
avgrq-sz	平均每次IO操作的数据量(扇区数为单位)
avgqu-sz	平均等待处理的IO请求队列长度
await	平均每次IO请求等待时间(包括等待时间和处理时间，毫秒为单位)
svctm	平均每次IO请求的处理时间(毫秒为单位)
%util	采用周期内用于IO操作的时间比率，即IO队列非空的时间比率

2. top

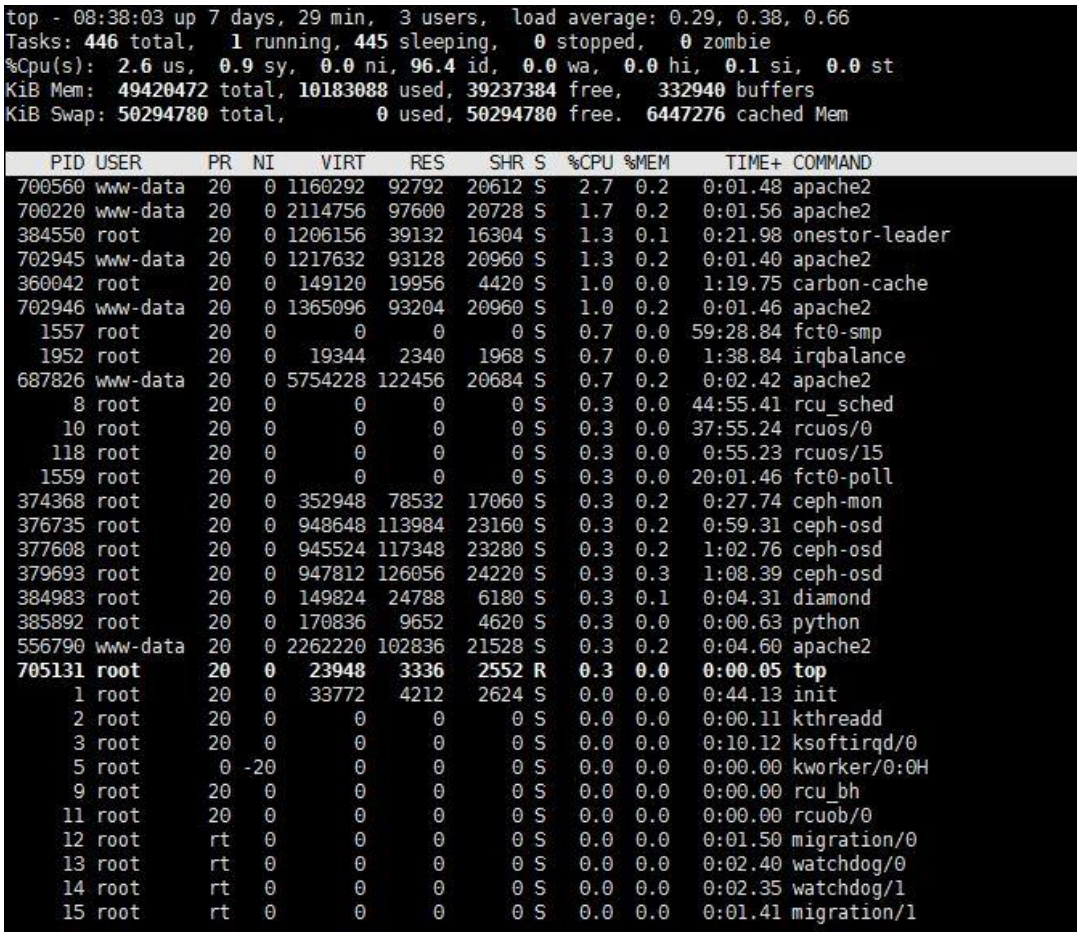
top 类似于 Windows 的资源管理器，能够实时监控系统各个进程的资源占用状况，该命令可以按 CPU 使用、内存使用和执行时间对任务进行排序；

经常关注的有如下几项：

- 看 load average 知道系统负载情况。
- 看 Tasks 看系统的任务数量，以及是否有僵尸进程 zombie。
- 看 CPU 消耗了多少。

通过对 CPU 或者内存使用情况来对进程进行排序，查看哪些进程导致了现在系统的问题（这个可以在 top 命令显示系统情况时，按大写的 F 或 O 键，选择 k 或者 n 来按照 cpu 和内存使用对进程排序显示。

top 显示界面如下：



- 各项输出项目的含义如下：前五行是系统整体的统计信息。
- 第一行是任务队列信息。其内容为，当前时间，系统运行时间，当前登录用户数，系统负载，即任务队列的平均长度，三个数值分别为 1 分钟、5 分钟、15 分钟前到现在的平均值。第二、三行为进程和 CPU 的信息。当有多个 CPU 时，这些内容可能会超过两行。内存中的内容被换出到交换区，而后又被换入到内存，但使用过的交换区尚未被覆盖，该数值即为这些内容已存在于内存中的交换区的大小。相应的内存再次被换出时可不必再对交换区写入。

统计信息区域的下方显示了各个进程的详细信息。

项目	含义
PID	进程id
RUSER	Real user name
UID	进程所有者的用户id
USER	进程所有者的用户名
VIRT	进程使用的虚拟内存总量，单位kb
RES	进程使用的、未被换出的物理内存大小，单位kb
SHR	共享内存大小，单位kb
%MEM	进程使用的物理内存百分比

%CPU	上次更新到现在的CPU时间占用百分比

可以通过快捷键来更改显示内容，按大写的“F”或“O”键，然后按 a-z 可以将进程按照相应的列进行排序。而大写的“R” 键可以将当前的排序倒转。

在 top 命令执行过程中可以使用以下交互命令。

项目	含义
q /Ctrl+C	退出程序
m	切换显示内存信息
t	切换显示进程和CPU状态信息
c	切换显示命令名称和完整命令行
M	根据驻留内存大小进行排序
P	根据CPU使用百分比大小进行排序
T	根据时间/累计时间进行排序

3. 其他查询命令

(1) lsblk

lsblk 命令查看硬盘容量、分区、使用，挂载信息。

```
root@node117:~# lsblk
NAME                      MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda                         8:0    0 279.4G  0 disk
├─sda1                      8:1    0   244M  0 part /boot
├─sda2                      8:2    0     1K  0 part
└─sda5                      8:5    0 279.1G  0 part
  ├─node117--vg-root (dm-0)   252:0    0 231.2G  0 lvm  /
  └─node117--vg-swap_1 (dm-1) 252:1    0    48G  0 lvm  [SWAP]
sdb                         8:16   0   1.1T  0 disk
sdc                         8:32   0 558.9G  0 disk
├─sdc1                      8:33   0 548.9G  0 part /var/lib/ceph/osd/ceph-2
└─sdc2                      8:34   0    10G  0 part
sdd                         8:48   0 558.9G  0 disk
├─sdd1                      8:49   0 548.9G  0 part /var/lib/ceph/osd/ceph-5
└─sdd2                      8:50   0    10G  0 part
sde                         8:64   0 558.9G  0 disk
├─sde1                      8:65   0 548.9G  0 part /var/lib/ceph/osd/ceph-8
└─sde2                      8:66   0    10G  0 part
sdf                         8:80   0 558.9G  0 disk
├─sdf1                      8:81   0 548.9G  0 part /var/lib/ceph/osd/ceph-11
└─sdf2                      8:82   0    10G  0 part
sdg                         8:96   0 558.9G  0 disk
├─sdg1                      8:97   0 548.9G  0 part /var/lib/ceph/osd/ceph-14
└─sdg2                      8:98   0    10G  0 part
sdh                         8:112  0 558.9G  0 disk
├─sdh1                      8:113  0 548.9G  0 part /var/lib/ceph/osd/ceph-17
└─sdh2                      8:114  0    10G  0 part
```

如上图 NAME 列出所有硬盘及分区，SIZE 显示硬盘总容量及分区大小，TYPE 显示硬盘及分区类型，MOUTPOINT 显示文件系统挂载点；sda 为系统盘，大小 279.4G，6 个 558.9G 大小的硬盘作为 OSD 分别进行挂载，日志分区大小为 10G。

(2) mount

查看集群内所有被挂载的文件系统及类型。

```
root@node117:~# mount
/dev/mapper/node117--vg-root on / type ext4 (rw,errors=remount-ro)
proc on /proc type proc (rw,noexec,nosuid,nodev)
sysfs on /sys type sysfs (rw,noexec,nosuid,nodev)
none on /sys/fs/cgroup type tmpfs (rw)
none on /sys/fs/fuse/connections type fusectl (rw)
none on /sys/kernel/debug type debugfs (rw)
none on /sys/kernel/security type securityfs (rw)
udev on /dev type devtmpfs (rw,mode=0755)
devpts on /dev/pts type devpts (rw,noexec,nosuid,gid=5,mode=0620)
tmpfs on /run type tmpfs (rw,noexec,nosuid,size=10%,mode=0755)
none on /run/lock type tmpfs (rw,noexec,nosuid,nodev,size=5242880)
none on /run/shm type tmpfs (rw,nosuid,nodev)
none on /run/user type tmpfs (rw,noexec,nosuid,nodev,size=104857600,mode=0755)
none on /sys/fs/pstore type pstore (rw)
/dev/sda1 on /boot type ext2 (rw)
systemd on /sys/fs/cgroup/systemd type cgroup (rw,noexec,nosuid,nodev,none,name=systemd)
/dev/sdc1 on /var/lib/ceph/osd/ceph-2 type xfs (rw,noatime,inode64)
/dev/sdd1 on /var/lib/ceph/osd/ceph-5 type xfs (rw,noatime,inode64)
/dev/sde1 on /var/lib/ceph/osd/ceph-8 type xfs (rw,noatime,inode64)
/dev/sdf1 on /var/lib/ceph/osd/ceph-11 type xfs (rw,noatime,inode64)
/dev/sdg1 on /var/lib/ceph/osd/ceph-14 type xfs (rw,noatime,inode64)
/dev/sdh1 on /var/lib/ceph/osd/ceph-17 type xfs (rw,noatime,inode64)
```

(3) df -h

列出所有挂载的文件系统，显示挂载的文件系统的总容量、使用容量、可用容量、使用百分比以及挂载点。

```
root@node117:~# df -h
Filesystem                  Size  Used Avail Use% Mounted on
/dev/mapper/node117--vg-root  228G  6.2G  210G   3% /
none                        4.0K     0  4.0K   0% /sys/fs/cgroup
udev                         24G   12K   24G   1% /dev
tmpfs                       4.8G   23M  4.7G   1% /run
none                        5.0M     0  5.0M   0% /run/lock
none                         24G   12K   24G   1% /run/shm
none                        100M     0  100M   0% /run/user
/dev/sda1                   237M   38M  187M  17% /boot
/dev/sdc1                   549G   42M  549G   1% /var/lib/ceph/osd/ceph-2
/dev/sdd1                   549G   42M  549G   1% /var/lib/ceph/osd/ceph-5
/dev/sde1                   549G   42M  549G   1% /var/lib/ceph/osd/ceph-8
/dev/sdf1                   549G   42M  549G   1% /var/lib/ceph/osd/ceph-11
/dev/sdg1                   549G   42M  549G   1% /var/lib/ceph/osd/ceph-14
/dev/sdh1                   549G   43M  549G   1% /var/lib/ceph/osd/ceph-17
```

显示 6 个 OSD 被挂载，单盘 OSD 容量 549G，使用率 1%

(4) fdisk -l

查看节点硬盘、分区、大小以及使用情况。

```
root@node110:~# fdisk -l

Disk /dev/sda: 300.0 GB, 299966445568 bytes
255 heads, 63 sectors/track, 36468 cylinders, total 585871964 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 262144 bytes / 262144 bytes
Disk identifier: 0x0000476d

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *         512      500223      249856   83  Linux
/dev/sda2          500734   585871359   292685313    5  Extended
Partition 2 does not start on physical sector boundary.
/dev/sda5          500736   585871359   292685312   8e  Linux LVM

WARNING: GPT (GUID Partition Table) detected on '/dev/sdb'! The util fdisk doesn't support GPT. Use GNU Parted.


Disk /dev/sdb: 1200.2 GB, 1200210141184 bytes
255 heads, 63 sectors/track, 145917 cylinders, total 2344160432 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 262144 bytes / 262144 bytes
Disk identifier: 0x00000000

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1  2344160431  1172080215+  ee  GPT
Partition 1 does not start on physical sector boundary.

WARNING: GPT (GUID Partition Table) detected on '/dev/sdc'! The util fdisk doesn't support GPT. Use GNU Parted.
```

(5) free

free 命令查看节点总内存，已用内存，buffer，cache，Swap 占用情况。

```
root@node117:~# free -h
             total       used       free     shared    buffers     cached
Mem:           47G       6.0G        41G        21M       176M       4.0G
-/+ buffers/cache:       1.8G        45G
Swap:          47G         0B        47G
```

7.1.12 云原生引擎容器服务相关命令

说明

相关命令需要在云原生引擎组件虚拟机上运行。

1. 查询集群组件运行状态

kubectl 用于运维 Kubernetes 集群。需要查看系统中组件运行状态，或者查看部署状态时，可运行如下命令。

```
root@HZ-UIS01-CVK01:~# kubectl get pod -A
```

```
NAMESPACE     NAME                                        READY   STATUS             RESTARTS   AGE
default       abc-5d95465487-nvwb2                        0/1     ImagePullBackOff   0          25d
default       abcde-7dc6868b6c-jpjzt                      0/1     ImagePullBackOff   0          25d
kube-system   alertmanager-main-0                         2/2     Running            2          39d
kube-system   coredns-f6c5964ff-m2hjp                     1/1     Running            1          109d
kube-system   coredns-f6c5964ff-nbh6m                     1/1     Running            1          109d
kube-system   custom-metrics-apiserver-5c546bfdc6-jz6pd   1/1     Running            1          39d
kube-system   etcd-10.125.37.220                          1/1     Running            1          109d
kube-system   flannel-pktpk                               1/1     Running            3          109d
kube-system   galaxy-daemonset-vq2bg                      1/1     Running            3          109d
kube-system   kube-apiserver-10.125.37.220                1/1     Running            5          109d
kube-system   kube-controller-manager-10.125.37.220       1/1     Running            187        109d
kube-system   kube-proxy-pb5sn                            1/1     Running            1          109d
kube-system   kube-scheduler-10.125.37.220                1/1     Running            116        109d
kube-system   kube-state-metrics-6d5db45dc6-vlrlr         1/1     Running            1          39d
kube-system   metrics-server-v0.3.6-78c965c9b-47xfd       2/2     Running            2          109d
kube-system   node-exporter-5ppln                         1/1     Running            1          39d
kube-system   prometheus-k8s-0                            3/3     Running            6          39d
kube-system   prometheus-operator-668d49fb58-gtsmv        1/1     Running            1          39d
tke           influxdb-0                                  1/1     Running            5          109d
tke           tke-application-api-577d84954b-96lkd        1/1     Running            6          109d
tke           tke-application-api-577d84954b-lljgr        1/1     Running            5          109d
tke           tke-application-controller-674f9b7854-2bghb 1/1     Running            73         109d
tke           tke-application-controller-674f9b7854-l9mmp 1/1     Running            75         109d
tke           tke-auth-api-77f9d7b9b5-s497k               1/1     Running            5          53d
tke           tke-auth-controller-5586d59cd4-5bxt4        1/1     Running            73         109d
tke           tke-auth-controller-5586d59cd4-pl494        1/1     Running            71         109d
tke           tke-gateway-8jrnv                           1/1     Running            6          53d
tke           tke-logagent-api-86cf899646-frcvj           1/1     Running            5          109d
tke           tke-logagent-api-86cf899646-w7scf           1/1     Running            2          109d
tke           tke-logagent-controller-6c7476b7d5-6btj2    1/1     Running            73         109d
tke           tke-logagent-controller-6c7476b7d5-q7m4q    1/1     Running            74         109d
tke           tke-monitor-api-84b9c864f8-b8zzt            1/1     Running            7          109d
tke           tke-monitor-api-84b9c864f8-k4tmh            1/1     Running            7          109d
tke           tke-monitor-controller-846cb6f76d-frlqd     1/1     Running            80         53d
tke           tke-notify-api-6969d6979b-slprd             1/1     Running            7          53d
tke           tke-notify-api-6969d6979b-z76sj             1/1     Running            7          53d
tke           tke-notify-controller-654cdf6575-2vbmh      1/1     Running            105        109d
tke           tke-notify-controller-654cdf6575-hq657      1/1     Running            108        109d
tke           tke-platform-api-5c67698f89-mnbmn           1/1     Running            3          109d
tke           tke-platform-api-5c67698f89-n7tq8           1/1     Running            5          109d
tke           tke-platform-controller-84879db4f5-xn247    1/1     Running            98         98d
tke           tke-registry-api-6649bc6f9-d5nmn            1/1     Running            6          109d
tke           tke-registry-api-6649bc6f9-q45gh            1/1     Running            6          109d
tke           tke-registry-controller-7d5bfdb796-jql4x    1/1     Running            79         109d
tke           tke-registry-controller-7d5bfdb796-q68t2    1/1     Running            78         109d
```

各列解释如下：

项目	含义
NAMESPACE	Pod所在命名空间
NAME	Pod名称
READY	当前状态，健康容器数量/运行容器数量
STATUS	Pod状态，包括Pending-挂起、Running-运行、Succeeded-成功退出、Failed-失败、Unknow-未知、XXBackoff-退避重启
RESTARTS	重启次数
AGE	总运行时长

若不为 Running 状态表示该组件有异常情况发生。

2. 查询集群组件日志

集群组件均以 Pod 形式运行于 Kubernetes 集群中，故使用 kubectl 查看其日志，常用命令如下：

- 查看 Pod 全部日志：`kubectl logs (NAME) [-c CONTAINER]`，如：`kubectl logs nginx`
- 跟随 Pod 全部日志：`kubectl logs (NAME) [-c CONTAINER] -f`，如：`kubectl logs nginx -f`
- 查看 Pod 最新日志：`kubectl logs (NAME) [-c CONTAINER] -tail=N`，如：`kubectl logs nginx -tail=100`

3. 重启集群组件

集群组件均以 Pod 形式运行于 Kubernetes 集群中，故使用 kubectl 重启组件，命令如下：

```
kubectl delete pod [-n NAMESPACE] (NAME)
```

例如重启 tke 命名空间下的 abc 容器：

```
kubectl delete pod -n tke abc
```

7.2 Linux常用命令

7.2.1 vi 文件编辑工具使用介绍

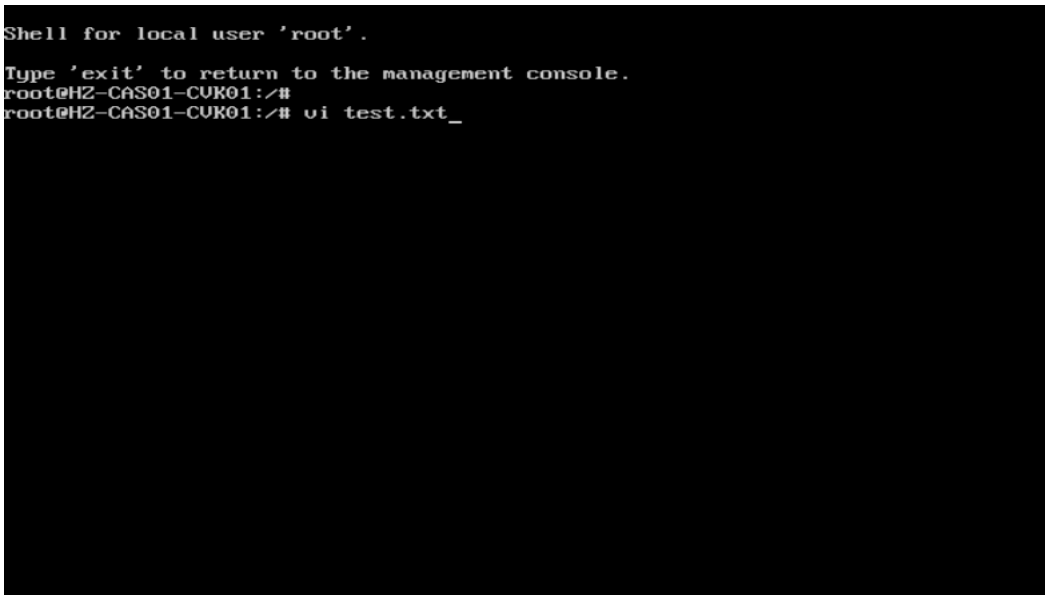
Linux 操作系统中，如果要新建文件或者编辑文件内容时，需要通过 vi 或者 vim 工具进行操作。该命令的使用频率非常高，因此必要要掌握。

vi 工具中需要理解一般模式和编辑模式，以及模式之间的切换方式。

如下以创建“test.txt”文件，文件内容为“123456”为例进行介绍。

1. 执行 vi 命令

在 Linux 的命令行窗口中输入“vi test.txt”命令，此时“test.txt”文件可以是已存在，也可以是未存在。如果该文件已存储，则可以通过 vi 工具修改内容；如果该文件不存在，则可以新建该文件。



2. 进入一般模式

执行完 vi 命令后，随即进入 vi 的一般模式。由于该文件是不存在的，已经执行该命令后，文件内容显示为空。

在一般模式中可以输入 vi 工具定义的按键命令，但是在该模式下不可以编辑文件内容。

3. 进入编辑模式

文件内容时在 vi 工具的编辑模式下进行编辑的。

在一般模式中直接按“i”、“o”、“a”键进入编辑模式。如下图所示：

4. 进入编辑模式

在编辑模式下，输入文件的内容。

5. 回到一般模式

在编辑模式完成文件的编写，需要回到一般模式。方法是在编辑模式下，按“ESC”。

6. 保存退出

回到一般模式后，需要保存文件并退出 vi，此时需要在一般模式下输入“:”（冒号）。

然后输入“wq”命令（write and quit），即可保存文件内容，并退出 vi。

“test.txt”文件创建完成后，执行 ls 命令即可查看到新创建的文件。

7.2.2 基本命令

1. 显示当前工作目录-pwd

pwd(Print Working Directory)用来打印当前工作目录。

```
root@HZ-UIS01-CVK01:~# pwd
/root
```

2. 显示文档信息-ls

ls(list)用来打印当前目录的文档信息。

```
# ls [-aAdfFhilnrRSt] 目录名称
选项与参数：
-a ：全部的文档，连同隐藏文档档一起列出来(常用)
-A ：全部的文档，连同隐藏档，但不包括.不..这两个目录
-d ：仅列出目录本身，而不是列出目录内的文档数据(常用)
-h ：将档案容量以人类较易读的方式(例如 GB，KB 等等)列出来
-i ：列出 inode 号码
-r ：将排序结果反向输出，例如：原本文档名由小到大，反向则为由大到小
-R ：连同子目录内容一起列出来
-S ：以档案容量大小排序，而不是用档名排序
-t ：依时间排序，而不是用档名
```

举例：

```
root@HZ-UIS01-UIS Manager:~# ls -al
total 44
drwx------  5 root root 4096 May 23 15:33 .
drwxr-xr-x 24 root root 4096 May 13 09:47 ..
-rw-------  1 root root  847 Jan  1 12:35 .bash_history
```

```
-rw-r--r--  1 root root 3106 Apr 19  2012 .bashrc
drwx------  2 root root 4096 May 17 17:23 .cache
-rw-r--r--  1 root root    8 May 23 15:33 UIS.conf
drwxr-xr-x  2 root root 4096 May 23 15:32 h3c
-rw-r--r--  1 root root  140 Apr 19  2012 .profile
drwxr-xr-x  2 root root 4096 May 22 09:50 .ssh
-rw-------  1 root root 4962 May 23 15:33 .viminfo
```

3. 变换目录-cd

```
cd(Change Directory)用来变换工作目录。
.  代表此层目录
.. 代表上一层目录
-  代表前一个工作目录
~ 代表『目前用户』所在的家目录
~account 代表 account 这个用户的家目录
```

举例：

```
root@HZ-UIS01-CVK01:/# cd ~root
#表示进入到 root 用户的家目录
root@HZ-UIS01-CVK01:~# cd ~
#表示回到自己的家目录
root@HZ-UIS01-CVK01:~# cd
#也表示回到自己的家目录
root@HZ-UIS01-CVK01:~# cd ..
#表示进入当前目录的上一层目录
root@HZ-UIS01-CVK01:/# cd -
#表示回到刚才的目录
root@HZ-UIS01-CVK01:~# cd /root
#表示进入“/root”目录
root@HZ-UIS01-CVK01:~# cd ../root
#表示进入上层目录下的 root 子目录
```

4. 创建新目录-mkdir

mkdir(make directory)创建新的目录。

```
# mkdir [-mp] 目录名称
选项与参数：
-m ：配置目录权限
-p ：将所需要的目录递归建立起来
```

举例：

```
root@HZ-UIS01-UIS Manager:~# ls
root@HZ-UIS01-UIS Manager:~# mkdir h3c
root@HZ-UIS01-UIS Manager:~# ls
h3c
root@HZ-UIS01-UIS Manager:~#
```

5. 复制文档或者目录-cp

cp(copy)用来复制文档或者目录。

```
# cp [-adfilprsu] 源文件(source) 目标文件(destination)
# cp [options] source1 source2 source3 .... 目标目录
选项与参数：
```

-a ：相当于-pdr 的意思(常用)
-f ：为强制(force)的意思
-i ：若目标文件已经存在时，在覆盖时会先询问动作的进行(常用)
-p ：连同文档的属性一起复制过去，而非使用默认属性(备份常用)；
-r ：递归持续复制，用于目录的复制行为；(常用)
注意：如果源文档有两个以上，则最后一个目的文件一定要是目录。

举例：

```
#复制文件
root@HZ-UIS01-UIS Manager:~# ls
UIS.conf
root@HZ-UIS01-UIS Manager:~# cp UIS.conf UIS.conf.bak
root@HZ-UIS01-UIS Manager:~# ls
UIS.conf  UIS.conf.bak
root@HZ-UIS01-UIS Manager:~#

#复制目录
root@HZ-UIS01-UIS Manager:~# ls
h3c
root@HZ-UIS01-UIS Manager:~# cp -rf h3c h3c.bak
root@HZ-UIS01-UIS Manager:~# ls
h3c  h3c.bak
root@HZ-UIS01-UIS Manager:~#
```

6. 远程拷贝文件-scp

scp 是 secure copy 的简写，用于在 Linux 下进行远程拷贝文件的命令，和它类似的命令有 cp，不过 cp 只是在本机进行拷贝不能跨服务器，而且 scp 传输是加密的。可能会稍微影响一下速度。当你服务器硬盘变为只读 read only system 时，用 scp 可以帮你把文件移出来。

```
#scp [参数] [原路径] [目标路径]
选项与参数：
-1  强制 scp 命令使用协议 ssh1
-2  强制 scp 命令使用协议 ssh2
-4  强制 scp 命令只使用 IPv4 寻址
-6  强制 scp 命令只使用 IPv6 寻址
-B  使用批处理模式（传输过程中不询问传输口令或短语）
-C  允许压缩。（将-C 标志传递给 ssh，从而打开压缩功能）
-p 保留原文件的修改时间，访问时间和访问权限。
-q  不显示传输进度条。
-r  递归复制整个目录。
-v 详细方式显示输出。scp 和 ssh(1)会显示出整个过程的调试信息。这些信息用于调试连接，验证和配置问题。
-c cipher  以 cipher 将数据传输进行加密，这个选项将直接传递给 ssh。
-F ssh_config  指定一个替代的 ssh 配置文件，此参数直接传递给 ssh。
-i identity_file  从指定文件中读取传输时使用的密钥文件，此参数直接传递给 ssh。
-l limit  限定用户所能使用的带宽，以 Kbit/s 为单位。
-o ssh_option  如果习惯于使用 ssh_config(5)中的参数传递方式，
-P port  注意是大写的 P, port 是指定数据传输用到的端口号
-S program  指定加密传输时所使用的程序。此程序必须能够理解 ssh(1)的选项。
```

举例：

```
root@HZ-UIS01-CVK01:~# scp UIS-E0218H06-Upgrade.tar.gz HZ-UIS01-CVK02:/root
```

```
UIS-E0218H06-Upgrade.tar.gz                                                    100%
545MB  90.8MB/s   00:06
root@HZ-UIS01-CVK01:~#
```

7. 删除文档或目录-rm

rm(remove)用来删除文档或目录。

```
# rm [-fir] 文件或目录
选项与参数：
-f ：就是 force 的意思，忽略不存在的文档，不会出现警告信息
-i ：互动模式，在删除前会询问使用者是否操作
-r ：递归删除。常用在目录的删除！这是非常危险的选项！！！
```

举例：

```
root@HZ-UIS01-UIS Manager:~# ls
h3c
root@HZ-UIS01-UIS Manager:~# rm -rf h3c
root@HZ-UIS01-UIS Manager:~# ls
root@HZ-UIS01-UIS Manager:~#
```

8. 移动文档与目录或更名-mv

mv(move)用来移动文档与目录或更名。

```
# mv [-fiu] source destination
# mv [options] source1 source2 source3 .... directory
选项与参数：
-f ：force 强制的意思，如果目标档案已经存在，不会询问而直接覆盖
-i ：若目标文档已经存在时，就会询问是否覆盖
-u ：若目标档案已经存在，且 source 比较新，才会更新(update)
```

举例：

```
root@HZ-UIS01-UIS Manager:~# ls
UIS.conf
root@HZ-UIS01-UIS Manager:~# mv UIS.conf UIS.conf.bak
root@HZ-UIS01-UIS Manager:~# ls
UIS.conf.bak
root@HZ-UIS01-UIS Manager:~#
```

9. 压缩与打包文档-tar

```
# tar [-j|-z] [cv] [-f 文档名] filename... <==打包与压缩
# tar [-j|-z] [xv] [-f 文档名] [-C 目录] <==解压缩
选项不参数：
-c ：建立打包文档
-t ：查看打包文档的内容含有哪些文档名
-x ：解压缩的功能，可以搭配-C （大写） 在特定目录解开
特别注意的是：-c，-t，-x 不可同时出现在一串指令中
-j ：透过 bzip2 的支持进行压缩/解压缩：此时档名最好为 *.tar.bz2
-z ：透过 gzip 的支持进行压缩/解压缩：此时档名最好为 *.tar.gz
-v ：在压缩/解压缩的过程中，将正在处理的文件名显示出来
-f filename：-f 后面要立刻接要被处理的文档名
-C 目录 ：这个选项用在解压缩，若要在特定目录解压缩，可以使用这个选项
```

举例：

```
#打包并压缩
```

```
root@HZ-UIS01-UIS Manager:~# ls
UIS.conf  UIS.conf-01  UIS.conf-02
root@HZ-UIS01-UIS Manager:~# tar -czvf UIS.tar.gz UIS.conf*
UIS.conf
UIS.conf-01
UIS.conf-02
root@HZ-UIS01-UIS Manager:~# ls
UIS.conf  UIS.conf-01  UIS.conf-02  UIS.tar.gz

#解压缩
root@HZ-UIS01-UIS Manager:~# ls
UIS.tar.gz
root@HZ-UIS01-UIS Manager:~# tar -xzvf UIS.tar.gz
UIS.conf
UIS.conf-01
UIS.conf-02
root@HZ-UIS01-UIS Manager:~# ls
UIS.conf  UIS.conf-01  UIS.conf-02  UIS.tar.gz
```

7.2.3 系统相关命令

1. 查看系统内核-uname

```
# uname [-asrmpi]
选项与参数：
-a：所有系统相关的信息，包括底下的数据都会被列出来
-s：系统内核名称
-r：  内核的版本
-m：本系统的硬件名称，例如 i686 戒 x86_64 等
-p：  CPU 的类型，与-m 类似，另是显示的是 CPU 的类型
-i ：  硬件的平台(x86)
```

举例：

```
root@ZJ-UIS-001:~# uname -a
Linux ZJ-UIS-001 4.1.0-generic #1 SMP Wed Nov 9 02:04:23 CST 2016 x86_64 x86_64 x86_64
GNU/Linux
```

2. 查看系统启动时间与负荷-uptime

举例：

```
root@HZ-UIS01-UIS Manager:~# uptime
 17:54:04 up 3 days, 23:28,  1 user,  load average: 0.08, 0.12, 0.13
```

3. 查看系统资源变化-vmstat

```
# vmstat [-a] [延迟 [总计监测次数]] <==CPU/内存等信息
# vmstat [-fs] <==内存相关
# vmstat [-S 单位] <==设定显示数据的单位
# vmstat [-d] <==与磁盘有关
# vmstat [-p 分区] <==与磁盘有关
选项与参数：
-a ：使用 inactive/active(活跃与否) 取代 buffer/cache 的内存输出信息
```

```
-f ：将开机到目前为止，系统复制(fork)的进程数
-s ：将一些事件 (开机至目前为止) 导致的内存变化情况列表说明
-S ：后面可以接单位，让显示的数据有单位。例如 K/M 取代 bytes 的容量
-d ：列出磁盘的读写总量统计表
-p ：后面列出分区，可显示该分区的读写总量统计表
```

举例：

```
root@HZ-UIS01-CVK01:~# vmstat 1 5
procs ---------------memory------------------ -----swap---- -----io---- ----system--
-----cpu--------
 r     b     swpd      free        buff      cache      si   so    bi    bo       in      cs
us   sy  id  wa
 1     0           0  60402384   58716 1712736    0    0    15       6       87     116   1   0
98   0
 0     0           0  60402500   58716 1712736    0    0     1        0      631   1051  0   0
100   0
 0     0           0  60402608   58756 1712752    0    0     0   840   1444   1640  2   0   98
0
 0     0           0  60403360   58756 1712760    0    0     2      33     991   1346  0   0 100
0
 2     0           0  60400944   58780 1712784    0    0     0      60   2225   1682  0   0   99
0
```

(1) 内存字段（procs）：

- r：等待运行中的程序数量；b：不可被唤醒的程序数量；这两项越多代表系统越忙碌。
- 内存字段（memory）：
  - swpd：虚拟内存被使用的容量；free：未被使用的内存容量；buff：用于缓冲存储器。
  - cache：用于高速缓存。

(2) 内存置换空间（swap）：

- si：由磁盘中将程序取出的量；so：由于内存不足而将没用到的程序写入到磁盘的 swap 容量；如果 si/so 的数值太大，表示内存内的数据常常在磁盘与主存储器之间传来传去，系统效率很低。
- 磁盘读写（io）：
  - bi：由磁盘写入的区块数量。
  - bo：写入到磁盘去的区块数量。如果这部分的数值越高，代表系统的 I/O 非常忙绿。

(3) 系统（system）：

- in：每秒被中断的程序次数；cs：每秒进行的事件切换次数；这两个数值越大，表示系统与接口设备的沟通发次频繁。这些接口包括磁盘、网卡、时钟等。
- CPU：
  - us：非核心层的 CPU 使用状态。
  - sy：核心层所使用的 CPU 状态；id：闲置的状态。
  - wa：等待 I/O 所耗费的 CPU 状态。
  - st：被虚拟机（Virtual machine）所盗用的 CPU 使用状态（2.6.11 以后支持）

4. 查看设备负载-iostat

iostat 用于输出 CPU 和磁盘 I/O 相关的统计信息。

#iostat[参数][时间][次数]

选项与参数：

-c 仅显示 CPU 统计信息，与-d 选项互斥。

-d 仅显示磁盘统计信息，与-c 选项互斥。

-k 以 KB 为单位显示每秒的磁盘请求数,默认单位块.

-m 以 MB 为单位显示

-N 显示磁盘阵列(LVM) 信息

-n 显示 NFS 使用情况

-p[磁盘] 与-x 选项互斥，用于显示块设备及系统分区的统计信息。也可以在-p 后指定一个设备名,如:# iostat -p /dev/sda

-t 在输出数据时，打印搜集数据的时间

-x 显示详细信息

-V 显示版本信息

说明：

- avg-cpu 段：
  - %user: 在用户级别运行所使用的 CPU 的百分比。
  - %nice: nice 操作所使用的 CPU 的百分比。
  - %sys: 在系统级别(kernel)运行所使用 CPU 的百分比。
  - %steal：管理程序维护另一个虚拟处理器时，虚拟 CPU 的无意识等待时间百分比。
  - %iowait: CPU 等待硬件 I/O 时，所占用 CPU 百分比。
  - %idle: CPU 空闲时间的百分比。
- Device 段：
  - tps: 每秒钟发送到的 I/O 请求数。
  - Blk_read /s: 每秒读取的 block 数。
  - Blk_wrtn/s: 每秒写入的 block 数。
  - Blk_read: 读入的 block 总数。
  - Blk_wrtn: 写入的 block 总数。

说明

- 如果%iowait 的值过高，表示硬盘存在 I/O 瓶颈，%idle 值高，表示 CPU 较空闲。
- 如果%idle 值高但系统响应慢时，有可能是 CPU 等待分配内存，此时应加大内存容量。
- %idle 值如果持续低于 10，那么系统的 CPU 处理能力相对较低，表明系统中最需要解决的资源是 CPU。

---

iostat 输出项目说明：

- Blk_read：读入块的当总数。
- Blk_wrtn 写入块的总数。
- kB_read/s：每秒从驱动器读入的数据量，单位为 K。

- kB_wrtn/s：每秒向驱动器写入的数据量，单位为 K。
- kB_read：读入的数据总量，单位为 K。
- kB_wrtn：写入的数据总量，单位为 K。
- rrqm/s：将读入请求合并后，每秒发送到设备的读入请求数。
- wrqm/s：将写入请求合并后，每秒发送到设备的写入请求数。
- r/s：每秒发送到设备的读入请求数。
- w/s：每秒发送到设备的写入请求数。
- rsec/s：每秒从设备读入的扇区数。
- wsec/s：每秒向设备写入的扇区数。
- rkB/s：每秒从设备读入的数据量，单位为 K。
- wkB/s：每秒向设备写入的数据量，单位为 K。
- avgrq-sz：发送到设备的请求的平均大小，单位是扇区。
- avgqu-sz：发送到设备的请求的平均队列长度。
- await：I/O 请求平均执行时间，包括发送请求和执行的时间，单位是毫秒。
- svctm：发送到设备的 I/O 请求的平均执行时间，单位是毫秒。
- %util：在 I/O 请求发送到设备期间，占用 CPU 时间的百分比，用于显示设备的带宽利用，当这个值接近 100%时，表示设备带宽已经占满。

举例：

```
root@HZ-UIS01-CVK01:~# iostat
Linux 3.13.6 (HZ-UIS01-CVK01)    12/16/2015       _x86_64_         (24 CPU)
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
                  20.48    0.00       3.48         0.23       0.00      75.80

Device:             tps     kB_read/s     kB_wrtn/s     kB_read       kB_wrtn
sda                10.17          1.76          269.57         1309400       201017740
sdb                16.43        181.78          202.21         135552881   150792613
```

执行“iostat -d -x -m /dev/sdb 1 5”命令查看“/dev/sdb”设备的详细信息。

```
root@HZ-CAS01-CVK01:~# iostat -d -x -m /dev/sdb 1 5
Linux 3.13.6 (HZ-CAS01-CVK01)   12/16/2015      _x86_64_       (24 CPU)

Device:         rrqm/s   wrqm/s     r/s     w/s    rMB/s    wMB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sdb               0.00     9.32    6.61    9.79     0.18     0.20    46.71     0.05    2.78    4.70    1.48   1.47   2.41

Device:         rrqm/s   wrqm/s     r/s     w/s    rMB/s    wMB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sdb               0.00     0.00    0.00    1.00     0.00     0.01    16.00     0.00    0.00    0.00    0.00   0.00   0.00

Device:         rrqm/s   wrqm/s     r/s     w/s    rMB/s    wMB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sdb               0.00     0.00    1.00    9.00     0.00     0.04     8.40     0.00    0.40    0.00    0.44   0.40   0.40

Device:         rrqm/s   wrqm/s     r/s     w/s    rMB/s    wMB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sdb               0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00

Device:         rrqm/s   wrqm/s     r/s     w/s    rMB/s    wMB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
sdb               0.00     0.00    1.00    8.00     0.00     0.04     8.56     0.00    0.00    0.00    0.00   0.00   0.00
```

5. 测试磁盘读写性能-dd

```
# dd [option]
```

选项与参数：

- if=file：输入文件名，缺省为标准输入。
- of=file：输出文件名，缺省为标准输出。

- ibs=bytes：一次读 bytes 个字节（即一个块大小为 bytes 个字节）。
- obs=bytes：一次写 bytes 个字节（即一个块大小为 bytes 个字节）。
- bs=bytes：同时设置读写块的大小为 bytes ，可代替 ibs 和 obs。
- cbs=bytes：一次转换 bytes 个字节，即转换缓冲区大小。
- skip=blocks：从输入文件开头跳过 blocks 个块后再开始复制。
- seek=blocks：从输出文件开头跳过 blocks 个块后再开始复制。（通常只有当输出文件是磁盘或磁带时才有效）
- count=blocks：仅拷贝 blocks 个块，块大小等于 ibs 指定的字节数
- conv=ASCII：把 EBCDIC 码转换为 ASCII 码。
- conv=ebcdic：把 ASCII 码转换为 EBCDIC 码。
- conv=ibm：把 ASCII 码转换为 alternate EBCDIC 码。
- conv=block：把变动位转换成固定字符。
- conv=ublock：把固定位转换成变动位。
- conv=uUISe：把字母由小写转换为大写。
- conv=lUISe：把字母由大写转换为小写。
- conv=notrunc：不截短输出文件。
- conv=swab：交换每一对输入字节。
- conv=noerror：出错时不停止处理。
- conv=sync：把每个输入记录的大小都调到 ibs 的大小（用 NUL 填充）。

需要注意的是，指定数字的地方若以下列字符结尾乘以相应的数字：b=512, c=1, k=1024, w=2, xm=number m，kB=1000，K=1024，MB=1000*1000，M=1024*1024，GB=1000*1000*1000，G=1024*1024*1024

6. 查看内存-free

```
# free [-b|-k|-m|-g] [-t]
```

选项与参数：

- -b ：直接输入 free 时，显示的单位是 Kbytes，可使用 b(bytes), m(Mbytes) k(Kbytes), 及 g(Gbytes) 来显示。
- -t ：在输出的最终结果，显示物理内存与 swap 的总量。

举例：

```
root@HZ-UIS01-CVK01:~# free
                     total           used             free      shared     buffers     cached
Mem:  65939360    4208888   61730472             0       83224     277944
-/+ buffers/cache:   3847720   62091640
Swap: 10772220               0    10772220
```

7.2.4 用户相关命令

1. 增加用户群组-groupadd

```
# groupadd [-g gid] groupname
```

选项与参数：

-g ： 后面接某个特定的 GID

举例：

```
root@HZ-UIS01-CVK01:~# groupadd -g 1000 it
root@HZ-UIS01-CVK01:/etc# more /etc/group | grep it
it:x:1000:
```

2. 删除用户群组-groupdel

```
# groupdel groupname
```

举例：

```
root@HZ-UIS01-CVK01:/etc# more /etc/group | grep it
it:x:1000:
root@HZ-UIS01-CVK01:/etc# groupdel it
root@HZ-UIS01-CVK01:/etc# more /etc/group | grep it
root@HZ-UIS01-CVK01:/etc#
```

3. 增加用户-useradd

```
# useradd [-u UID] [-g 初始群组] [-G 次要群组] [-m/M] [-d 家目录绝对路径] [-s shell] username
```

选项与参数：

- -u ： 后面接接的是 UID
- -g ： 后面接的是初始群组
- -G ： 后面接的组名则是这个账号还可以加入的群组
- -M ： 强制，不要建立用户家目录
- -m ： 强制，建立用户家目录
- -d ： 指定某个目录为家目录
- -s ： 后面接一个 shell ， 若没有设定则默认是 /bin/bash

举例：

```
root@HZ-UIS01-CVK01:~# useradd -u 1000 -g it  -m -d /home/it-user01 -s /bin/bash it-user01
root@HZ-UIS01-CVK01:~# more /etc/passwd | grep it-user01
it-user01:x:1000:1000::/home/it-user01:/bin/bash
root@HZ-UIS01-CVK01:~# ls /home/
it-user01
```

4. 删除用户-userdel

```
# userdel [-r] username
```

选项与参数：

-r ： 连同用户的家目录一起删除

举例：

```
root@HZ-UIS01-CVK01:~# userdel -r it-user01
root@HZ-UIS01-CVK01:~# more /etc/passwd | grep it-user01
root@HZ-UIS01-CVK01:~# ls /home
```

```
root@HZ-UIS01-CVK01:~#
```

5. 设置用户密码-passwd

```
passwd [-l] [-u] [--sdtin] [-S] [-n 日数] [-x 日数] [-w 日数] [-i 日期] username
```

选项与参数：

- -l ：是 Lock 的意思，会将/etc/shadow 第二栏最前面加上！是密码失效
- -u ：与-l 相对，是 Unlock 的意思
- -S ：列出密码相关参数，亦即 shadow 档案内的大部分信息
- -n ：后面接天数，多久不可修改密码天数
- -x ：后面接天数，多久内必须修改密码
- -w ：后面接天数，密码过期前的警告天数
- -i ：后面接『日期』，密码失效日期

举例：

```
root@HZ-UIS01-CVK01:~# more /etc/passwd | grep it-user01
it-user01:x:1000:1000::/home/it-user01:/bin/bash
root@HZ-UIS01-CVK01:~#
root@HZ-UIS01-CVK01:~# passwd it-user01
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

6. 切换用户-su

```
su [-lm] [-c 指令] [username]
```

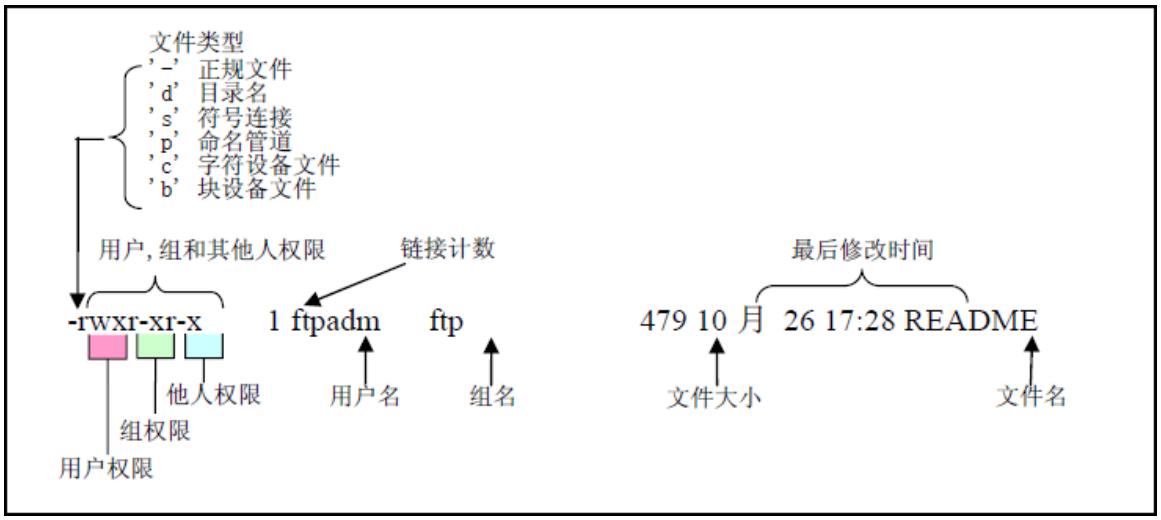
选项与参数：

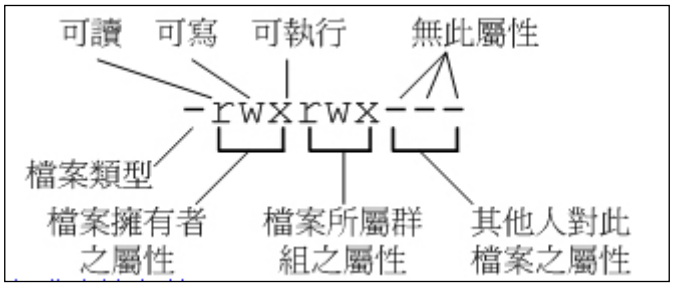
- - ：单纯使用“-”，如『 su - 』代表使用 login-shell 的变量文档读取方式来登入系统；
- 若使用者名称没有加上去，则代表切换为 root 的身份。
- -l ：与“-”类似，但后面需要加欲切换的使用者账号！也是 login-shell 的方式。
- -m ：-m 与-p 是一样的，表示『使用目前的环境设定，而不读取新使用者的配置文件』
- -c ：仅进行一次指令，所以 -c 后面可以加上指令。

举例：

```
root@HZ-UIS01-CVK01:~# su - it-user01
it-user01@HZ-UIS01-CVK01:~$ exit
logout
it-user01@HZ-UIS01-CVK01:~$ su - root
Password:
root@HZ-UIS01-CVK01:~#
```

7.2.5 文档属性相关命令





1. 更改文档的用户群组-chgrp

```
# chgrp [-R] 组名 目录/文件
```

选项与参数：

-R : 进行递归(recursive)的持续修改，即连同次目录下的所有文件、目录都更新成为这个群组。

举例：

```
root@HZ-UIS01-CVK01:/home/it-user01# ls -l
total 4
drwxr-xr-x 2 it-user01 it 4096 May 30 15:44 testFile
root@HZ-UIS01-CVK01:/home/it-user01# chgrp root testFile
root@HZ-UIS01-CVK01:/home/it-user01# ls -l
total 4
drwxr-xr-x 2 it-user01 root 4096 May 30 15:44 testFile
```

2. 更改文档的拥有者-chgrp

```
# chown [-R] 用户 文件或目录
# chown [-R] 用户:组名 文件或目录
```

选项与参数：

-R：进行递归(recursive)的持续变更，即连同次目录下的所有文件都修改。

举例：

```
root@HZ-UIS01-CVK01:/home/it-user01# ls -l
total 4
drwxr-xr-x 2 it-user01 it 4096 May 30 15:44 testFile
root@HZ-UIS01-CVK01:/home/it-user01# chown root:root testFile
root@HZ-UIS01-CVK01:/home/it-user01# ls -l
total 4
drwxr-xr-x 2 root root 4096 May 30 15:44 testFile
root@HZ-UIS01-CVK01:/home/it-user01#
```

3. 修改文档权限属性-chmod

```
# chmod [-R] xyz 文件或目录
```

选项与参数：

- xyz : 数字类型的权限属性，为 rwx 属性数值的相加
- -R : 进行递归(recursive)的持续变更，即连同次目录下的所有文件都会修改

举例：

```
root@HZ-UIS01-CVK01:/home/it-user01# ls -l
total 4
drwxr-xr-x 2 it-user01 it 4096 May 30 15:44 testFile
root@HZ-UIS01-CVK01:/home/it-user01# chmod 777 testFile
root@HZ-UIS01-CVK01:/home/it-user01# ls -l
total 4
drwxrwxrwx 2 it-user01 it 4096 May 30 15:44 testFile
root@HZ-UIS01-CVK01:/home/it-user01#
```

7.2.6 进程相关命令

1. 动态查看进程变化-top

```
# top [-d 数字] | top [-bnp]
```

选项与参数：

- -d ： 后面可以接秒数，就是整个程序画面更新的秒数。默认是 5 秒；
- -b ： 以批次的方式执行 top。通常会搭配数据流重导向来将批次的结果输出成为文档。
- -n ： 与-b 搭配，进行几次 top 的输出结果。
- -p ： 指定某些个 PID 来进行观察监测

在 top 执行过程当中可以使用的按键指令：

- ? ： 显示在 top 当中可以输入的按键指令
- P ： 以 CPU 的使用资源排序显示
- M ： 以 Memory 的使用资源排序显示
- N ： 以 PID 来排序
- T ： 由进程使用的 CPU 时间累积 (TIME+) 排序
- k ： 给某个 PID 一个信号
- r ： 给某个 PID 重新制订一个 nice 值
- q ： 离开 top 的按键

举例：

```
top - 17:40:48 up  2:13,  1 user,  load average: 0.45, 0.55, 0.66
```

```
Tasks: 257 total,   1 running, 256 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.6%us,  0.1%sy,  0.0%ni, 99.2%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  65939360k total,  5703848k used, 60235512k free,    85832k buffers
Swap: 10772220k total,        0k used, 10772220k free,  1746992k cached

  PID USER      PR  NI    VIRT    RES    SHR    S  %CPU  %MEM          TIME+      COMMAND
 4939   root       20   0   4583m   1.3g    4728   S        12          2.1        17:36.67
kvm
 4874   root      20   0   4520m  908m   4576   S         5        1.4       11:54.61    kvm
 4043   root       20   0     10.9g  402m    16m   S          1         0.6        13:43.34
java
 2370   root       20   0    23676   2168   1316   S          0         0.0         0:30.29
ovs-vswitchd
 3184   root       20   0    15972    744     544   S          0         0.0         0:04.78
irqbalance
      1    root       20   0    24456   2444   1344   S          0         0.0         0:04.07
init
      2    root      20   0         0        0        0   S        0        0.0         0:00.00
kthreadd
      3    root      20   0         0        0        0   S        0        0.0         0:00.07
ksoftirqd/0
      6    root     RT   0         0        0        0   S        0        0.0         0:00.00
migration/0
```

显示信息说明：

- 第一行：当前的时间；系统开机到目前为止经过的时间；已经登录系统的用户人数；系统在1,5,15 分钟的平均负载，越小代表系统越闲置，若高于 1 时需要注意系统的程序是否过于繁忙。
- 第二行：需要关注的是 zombie，如果该值不是 0，需要分析是哪个进程变成僵尸进程了。
- 第三行：显示 CPU 的整体负载。特别需要关注的是%wa，代表 I/O wait，通常系统变慢都是 I/O 产
- 生的问题较大。
- 第四行和第五行：显示当前物理内存和虚拟内存（Mem/Swap）的使用情况。需要关注的是 swap 的使用量要尽量的少。若果 swap 被大量使用，表明系统的物理内存实在不足。

Top 的下半部分，显示每个进程的资源情况，需要关注的是：

- PID：每个进程的 ID
- USER：进程的使用者
- PR：Priority，程序的优先级信息，越小越早被执行
- NI：nice 的简写，与 Priority 有关，也是越小越早被执行
- %CPU：CPU 的使用率
- %MEM：内存的使用率
- TIME+：CPU 使用的时间累加

举例：查看某单一进程的信息。

```
root@HZ-UIS01-CVK01:~# top -d 2 -p 4939

top - 08:59:13 up 17:31,  1 user,  load average: 0.75, 0.70, 0.58
```

```
Tasks:   1 total,   0 running,   1 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.1%us,  0.1%sy,  0.0%ni, 99.8%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  65939360k total,  6484728k used, 59454632k free,   229880k buffers
Swap: 10772220k total,        0k used, 10772220k free,  1995728k cached

  PID  USER      PR  NI    VIRT    RES    SHR  S  %CPU %MEM    TIME+         COMMAND
 4939  root          20   0  4583m   1.5g   4728  S          2        2.4    100:48.79   kvm
```

2. 静态查看进程-ps

```
# ps aux    <==观察系统所有的程序信息
# ps -lA      <==也是能够观察所有系统的信息
# ps axjf    <==连同部分程序树状态
```

选项与参数：

- -A：所有的进程均显示出来
- -a：不与 terminal 有关的所有进程
- -u：有效使用者相关的进程
- -x：通常与 a 这个参数一起使用，可列出较完整信息

输出格式：

- l：较长、较详细的将该 PID 的信息列出
- j：工作的格式
- -f：做一个更为完整的输出

举例：显示自己 bash 的进程。

```
root@HZ-UIS01-CVK01:~# ps -l
F  S   UID    PID    PPID   C  PRI   NI   ADDR   SZ        WCHAN  TTY          TIME      CMD
4  R      0 11338  32857   0    80    0   -          2102     -            pts/2        00:00:00
ps
4  S      0 32857  32797   0    80    0   -          5428     wait         pts/2        00:00:00
bash
```

使用 ps –l 仅列出与操作环境（bash）有关的程序而已，亦即最上层的父程序会是自己的 bash，从而延伸到 init 进程。

- F：表示进程的标志。4，表示程序的权限为 root；1，表示此子程序仅进行复制（fork）而没有实际执行（exec）。
- S：表示进程的状态。R，Running；S，Sleep；D，不可被唤醒的睡眠状态，通常该进程在等待 I/O。
- T，Stop；Z，Zombie 僵尸状态，程序已经终止但却无法移除到内存外。
- UID/PID/PPID：进程号
- C：CPU 的使用率。
- PRI/NI：Priority 和 Nice。
- ADDR/SZ/WCHAN：都与内存有关。ADDR 之处进程在内存的哪个部分，如果为 Running，一般显示“-”；SZ，表示此进程用掉多少内存；WCHAN，表示进行是否在运行中。
- TTY：登入者的终端机位置，若为远程登入则使用动态终端接口（pts/2）。
- CMD：command 的缩写。

举例：显示所有的进程。

```
root@HZ-UIS01-CVK01:~# ps aux
USER        PID  %CPU %MEM      VSZ    RSS     TTY      STAT START    TIME COMMAND
root              1      0.0         0.0    24572  2484        ?        Ss        11:20    0:04
/sbin/init
root              2      0.0         0.0          0        0         ?        S         11:20
0:00    [kthreadd]
root              3      0.0         0.0          0        0         ?        S         11:20
0:00    [ksoftirqd/0]
root              6      0.0         0.0          0        0         ?        S         11:20
0:00    [migration/0]
root              7      0.0         0.0          0        0         ?        S         11:20
0:00    [watchdog/0]
root              8      0.0         0.0          0        0         ?        S         11:20
0:00    [migration/1]
...................中间省略
root        55719     1.0          0.0   71272    3520        ?        Ss        17:42    0:00
sshd: root@pts/3
root        55752     8.6          0.0   21712    4204    pts/3       Ss        17:43    0:00  -bash
root        55927     0.0          0.0   16872    1284    pts/3        R+       17:43    0:00  ps
aux
root        62570     0.0          0.0          0         0         ?        S         14:43
0:00    [kworker/7:2]
root        62840     0.0          0.0          0         0         ?        S         16:40
0:00    [kworker/u:0]
```

举例：查看某进程信息。

```
root@HZ-UIS01-CVK01:~# ps -fu mysql
UID          PID    PPID  C    STIME TTY        TIME       CMD
mysql      3144          1  0       11:21      ?      00:00:46    /usr/sbin/mysqld
```

3. 进程管理-kill

```
# kill -signal PID
```

Signal 种类如下：

- 1 SIGHUP：启动被终止的程序，可让该 PID 重新读取自己的配置文件，类似重新启动。
- 9 SIGKILL：代表强制中断一个程序的运行。
- 15 SIGTERM：正常的结束程序来终止该程序。

7.2.7 网络相关命令

1. 查看网卡信息-ifconfig

举例：查看系统已启用网卡，命令“ifconfig”。

```
root@HZ-UIS01-CVK01:/etc/network# ifconfig
eth0      Link encap:Ethernet  HWaddr 2C:76:8A:5B:3F:A0
          UP BROADUIST MULTIUIST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:26 Memory:f6000000-f67fffff
eth1      Link encap:Ethernet  HWaddr 2C:76:8A:5B:3F:A4
```

```
          UP BROADUIST MULTIUIST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:28 Memory:f4800000-f4ffffff
……下面省略
```

查看系统所有网卡，命令“ifconfig -a”，该命令会将打印系统所有的网卡信息，包含的未启用的网卡。

举例：打印具体某网卡信息，命令“ifconfig 网卡名”。

```
root@HZ-UIS01-CVK01:/etc/network# ifconfig vswitch2
vswitch2  Link encap:Ethernet  HWaddr 2C:76:8A:5D:DF:A0
          inet addr:192.168.1.11  BUISt:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::2e76:8aff:fe5d:dfa0/64 Scope:Link
          UP BROADUIST RUNNING PROMISC MULTIUIST  MTU:1500  Metric:1
          RX packets:1134578 errors:0 dropped:7658 overruns:0 frame:0
          TX packets:1013948 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:165047129 (157.4 Mb)  TX bytes:111771007 (106.5 Mb)
```

举例：手动关闭网卡。

```
# ifconfig vswitch2 down
```

举例：手动启动网卡。

```
# ifconfig vswitch2 up
```

举例：手动配置网卡（重启网卡或重启系统后，该配置丢失）

```
# ifconfig vswitch2 192.168.2.12 netmask 255.255.255.0
```

举例：重启网卡

```
# /etc/init.d/networking restart
```

举例：为了永久的保存网卡的配置，需要使用 vi 工具修改网卡配置文件（/etc/network/interfaces），修改后需要执行重启网卡生效。

```
auto vswitch2
iface vswitch2 inet static
           address 192.168.1.11
           netmask 255.255.255.0
           network 192.168.1.0
           broadUISt 192.168.1.255
           gateway 192.168.1.254
           # dns-* options are implemented by the resolvconf package, if installed
           dns-nameservers 192.168.1.254
auto eth2
iface eth0 inet static
        address 0.0.0.0
        netmask 0.0.0.0
```

2. 查看物理网卡信息-ethtool

```
root@UIS-CVK02:~# ethtool eth1
Settings for eth1:
```

```
        Supported ports: [ TP ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                                100baseT/Half 100baseT/Full
                                                1000baseT/Half 1000baseT/Full
        Supported pause frame use: No
        Supports auto-negotiation: Yes
        Advertised link modes:  10baseT/Half 10baseT/Full
                                               100baseT/Half 100baseT/Full
                                               1000baseT/Half 1000baseT/Full
        Advertised pause frame use: Symmetric
        Advertised auto-negotiation: Yes
        Link partner advertised link modes:  10baseT/Half 10baseT/Full
                                                                    100baseT/Half
100baseT/Full
                                                                    1000baseT/Full
        Link partner advertised pause frame use: No
        Link partner advertised auto-negotiation: Yes
        Speed: 1000Mb/s
        Duplex: Full
        Port: Twisted Pair
        PHYAD: 1
        Transceiver: internal
        Auto-negotiation: on
        MDI-X: on
        Supports Wake-on: g
        Wake-on: g
        Current message level: 0x000000ff (255)
                               drv probe link timer ifdown ifup rx_err tx_err
        Link detected: yes
```

3. 查看网络信息-netstat

```
# netstat -[atunlp]
```

选项与参数：

- -a ：将目前系统上所有的联机、监听、Socket 数据都列出来
- -t ：列出 tcp 网络包数据
- -u ：列出 udp 网络包数据
- -n ：以端口号显示服务
- -l ：列出目前正在监听的服务
- -p ：列出该服务的进程 PID 信息

举例：查看使用 8080 端口服务的网络连接信息。

```
root@HZ-UIS01-CVK01:/etc/network# netstat -an | grep 8080
tcp6       0      0 :::8080                 :::*                    LISTEN
tcp6       0      0 192.168.1.11:8080       10.165.136.197:55954    ESTABLISHED
tcp6       0      0 192.168.1.11:8080       10.165.136.197:55989    TIME_WAIT
tcp6       0      0 192.168.1.11:8080       10.165.136.197:55990    FIN_WAIT2
tcp6       0      0 192.168.1.11:8080       192.168.1.211:53366     ESTABLISHED
tcp6       0      0 192.168.1.11:8080       192.168.1.211:54850     TIME_WAIT
```

举例：查看系统的路由信息。

```
root@HZ-UIS01-CVK01:/etc/network# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.1.254   0.0.0.0         UG        0 0          0
vswitch2
192.168.1.0     0.0.0.0         255.255.255.0   U         0 0          0 vswitch2
192.168.2.0     0.0.0.0         255.255.255.0   U         0 0          0 vswitch-storage
192.168.3.0     0.0.0.0         255.255.255.0   U         0 0          0 vswitch-app
```

4. 抓包-tcpdump

```
tcpdump
```

选项与参数：

- -a ： 将网络地址和广播地址转变成名字
- -d ： 将匹配信息包的代码以人们能够理解的汇编格式给出
- -dd ： 将匹配信息包的代码以 c 语言程序段的格式给出
- -ddd ： 将匹配信息包的代码以十进制的形式给出
- -e ： 在输出行打印出数据链路层的头部信息
- -t ： 在输出的每一行不打印时间戳
- -vv ： 输出详细的报文信息
- -c ： 在收到指定的包的数目后，tcpdump 就会停止
- -i ： 指定监听的网络接口
- -w ： 直接将包写入文件中，并不分析和打印出来

举例：

```
tcpdump -i vswitch2 -s 0 -w /tmp/test.cap host 200.1.1.1 &
```

5. 静态路由-route

举例：显示路由信息，命令“route -n”。

```
root@HZ-UIS01-CVK01:/etc/network# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.1.254   0.0.0.0         UG    100    0        0 vswitch2
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 vswitch2
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 vswitch-storage
192.168.3.0     0.0.0.0         255.255.255.0   U     0      0        0 vswitch-app
```

举例：增加访问“10.10.10.0/24”网络的静态路由信息。

```
# route add -net 10.10.10.0 netmask 255.255.255.0 gw 192.168.2.254
root@HZ-UIS01-CVK01:/etc/network#
root@HZ-UIS01-CVK01:/etc/network# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.1.254   0.0.0.0         UG    100    0        0 vswitch2
10.10.10.0      192.168.2.254   255.255.255.0   UG    0      0        0 vswitch-storage
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 vswitch2
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 vswitch-storage
```

```
192.168.3.0     0.0.0.0           255.255.255.0   U       0        0       0     vswitch-app
```

举例：删除静态路由信息。

```
# route del -net 10.10.10.0 netmask 255.255.255.0 gw 192.168.2.254
root@HZ-UIS01-CVK01:/etc/network# route -n
Kernel IP routing table
Destination     Gateway           Genmask          Flags  Metric  Ref     Use  Iface
0.0.0.0          192.168.1.254   0.0.0.0             UG    100       0        0     vswitch2
192.168.1.0     0.0.0.0            255.255.255.0   U       0         0        0     vswitch2
192.168.2.0     0.0.0.0          255.255.255.0   U       0        0       0     vswitch-storage
192.168.3.0     0.0.0.0            255.255.255.0   U       0        0       0     vswitch-app
```

通过执行命令生成的静态路由信息，仅保存在系统的内存中，如果需要永久生效，则需要将该命令添加到系统启动脚本中，使系统在启动过程就执行该命令。

操作方法为在 UIS 系统后台，通过 vi 工具，编辑“/etc/rc.local”文档，即“vi /etc/rc.local”。

在打开的文档中添加路由命令。添加完成后需要重启系统生效。

```
root@HZ-UIS01-CVK01:/etc/network# vi /etc/rc.local

#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

route add -net 192.168.5.0 netmask 255.255.255.0 gw 192.168.2.254

ulimit -s 10240
ulimit -c 1024
touch /var/run/h3c_UIS_cvk
/usr/bin/set-printk-console 2
exit 0
```

7.2.8 磁盘相关命令

1. 查看磁盘容量信息-df

```
# df [-ahikHTm] [目录或文件]
```

选项与参数：

- -a ： 列出所有的文件系统，包括系统特有的/proc 等文件系统
- -k ： 以 KBytes 的容量显示各文件系统
- -m ： 以 MBytes 的容量显示各文件系统
- -h ： 以人们较易阅读的 GBytes, MBytes, KBytes 等格式自行显示

- -H ：以 M=1000K 取代 M=1024K 的进位方式
- -T ：连同该分区的文件系统名称(例如 ext3)也列出
- -i ：不用硬盘容量，而以 inode 的数量来显示

举例：显示分区容量信息。

```
root@HZ-UIS01-CVK01:/etc/network# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1        28G  2.4G   25G   9% /
udev             32G  4.0K   32G   1% /dev
tmpfs            13G  396K   13G   1% /run
none            5.0M     0  5.0M   0% /run/lock
none             32G   17M   32G   1% /run/shm
/dev/sda6       241G   48G  181G  21% /vms
```

举例：包括显示分区的文件系统类型。

```
root@HZ-UIS01-CVK01:/etc/network# df -Th
Filesystem     Type          Size  Used Avail Use% Mounted on
/dev/sda1      ext4           28G  2.4G   25G   9% /
udev           devtmpfs       32G  4.0K   32G   1% /dev
tmpfs          tmpfs          13G  396K   13G   1% /run
none           tmpfs         5.0M     0  5.0M   0% /run/lock
none           tmpfs          32G   17M   32G   1% /run/shm
/dev/sda6      ext4          241G   48G  181G  21% /vms
```

2. 查看磁盘使用量信息-du

```
# du [-ahskm] 文件或目录名
```

选项与参数：

- -a ：列出所有的文件或目录容量，因为默认仅统计目录底下的文件量而已
- -h ：以人们较易读的容量格式 (G/M) 显示
- -s ：列出总量而已
- -S ：不包括子目录下的统计，与-s 有点差别
- -k ：以 KBytes 列出容量显示
- -m ：以 MBytes 列出容量显示

举例：

```
root@HZ-UIS01-CVK01:/vms# du -sh *
15G     images
11G     isos
16K     lost+found
3.4G    rhel-server-6.1-x86_64-dvd.iso
4.0K    share
4.0K    share-test
17G     templet
4.0K    test
```

3. 磁盘分区-fdisk

```
# fdisk [-l] 磁盘名称
```

选项与参数：

-l ：输出后面接的磁盘所有的分区内容。

若仅有“fdisk -l”时，则系统将会把整个系统内能够搜索到所有磁盘的分区都列出来。

举例：

```
root@HZ-UIS01-CVK01:~# fdisk -l
Disk /dev/sda: 300.0 GB, 299966445568 bytes
255 heads, 63 sectors/track, 36468 cylinders, total 585871964 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 262144 bytes / 262144 bytes
Disk identifier: 0x00051ce2

 Device Boot          Start              End           Blocks       Id    System
/dev/sda1   *             512      58593791     29296640    83    Linux
/dev/sda2         58594302   585871359   263638529       5    Extended
Partition 2 does not start on physical sector boundary.
/dev/sda5         58594304     80138751     10772224     82   Linux swap / Solaris
/dev/sda6         80139264   585871359   252866048     83   Linux

Disk /dev/sdb: 4294 MB, 4294967296 bytes
133 heads, 62 sectors/track, 1017 cylinders, total 8388608 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

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

举例：为磁盘创建分区。

```
root@HZ-UIS01-CVK01:~# fdisk /dev/sdb
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel with disk identifier 0xeb665aa3.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help): m
Command action
   a   toggle a bootable flag
   b   edit bsd disklabel
   c   toggle the dos compatibility flag
   d   delete a partition    “删除一个分区”
   l   list known partition types
   m   print this menu
   n   add a new partition “新建一个分区”
   o   create a new empty DOS partition table
   p   print the partition table “在屏幕打印分区信息”
   q   quit without saving changes “不保存配置信息离开 fdisk”
```

```
   s   create a new empty Sun disklabel
   t   change a partition's system id
   u   change display/entry units
   v   verify the partition table
   w   write table to disk and exit “保存配置信息并离开”
   x   extra functionality (experts only)

Command (m for help): p

Disk /dev/sdb: 4294 MB, 4294967296 bytes
133 heads, 62 sectors/track, 1017 cylinders, total 8388608 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xeb665aa3

Device Boot      Start         End      Blocks   Id  System

Command (m for help): n “新建一个分区”
Partition type:
   p   primary (0 primary, 0 extended, 4 free)
   e   extended
Select (default p): p “选择该分区为主分区”
Partition number (1-4, default 1): 1 “分区号”
First sector (2048-8388607, default 2048): “设置分区的起始扇区”
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-8388607, default 8388607): 4000000 “设置分区最后的扇区”
Command (m for help): n
Partition type:
   p   primary (1 primary, 0 extended, 3 free)
   e   extended
Select (default p): p
Partition number (1-4, default 2): 2
First sector (4000001-8388607, default 4000001): “设置分区的起始扇区”
Using default value 4000001
Last sector, +sectors or +size{K,M,G} (4000001-8388607, default 8388607): +500M “设置分区大小”
Command (m for help): p “打印分区信息”

Disk /dev/sdb: 4294 MB, 4294967296 bytes
133 heads, 62 sectors/track, 1017 cylinders, total 8388608 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xeb665aa3

Device Boot        Start            End          Blocks    Id   System
/dev/sdb1               2048     4000000     1998976+  83   Linux “新建的分区”
```

```
/dev/sdb2          4000001     5024000        512000   83   Linux “新建的分区”

Command (m for help): w “保存分区配置并离开”
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

举例：打印磁盘分区信息。

```
root@HZ-UIS01-CVK01:~# fdisk -l /dev/sdb

Disk /dev/sdb: 4294 MB, 4294967296 bytes
133 heads, 62 sectors/track, 1017 cylinders, total 8388608 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xeb665aa3

 Device Boot       Start              End          Blocks      Id   System
/dev/sdb1              2048       4000000     1998976+   83   Linux
/dev/sdb2          4000001     5024000          512000   83   Linux
```

4. 磁盘格式-mkfs

```
# mkfs [-t 文件系统格式] 磁盘名称
```

选项与参数：

```
-t ：后接文件系统格式，例如 ext2、ext3、ext4、ocfs2 等。
```

举例：将“/dev/sdb1”格式化为 ex3 格式是文件系统。

```
root@HZ-UIS01-CVK01:~# mkfs -t ext3 /dev/sdb1
mke2fs 1.42 (29-Nov-2011)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
125184 inodes, 499744 blocks
24987 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=515899392
16 block groups
32768 blocks per group, 32768 fragments per group
7824 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912

Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done
```

```
root@HZ-UIS01-CVK01:~#
```

举例：将“/dev/sdb1”格式化为 ocfs2 格式是文件系统。

```
root@HZ-UIS01-CVK01:~# mkfs -t ocfs2 /dev/sdb2
mkfs.ocfs2 1.6.3
Cluster stack: classic o2cb
Label:
Features: sparse backup-super unwritten inline-data strict-journal-super xattr
Block size: 1024 (10 bits)
Cluster size: 4096 (12 bits)
Volume size: 524288000 (128000 clusters) (512000 blocks)
Cluster groups: 17 (tail covers 5120 clusters, rest cover 7680 clusters)
Extent allocator size: 2097152 (1 groups)
Journal size: 16777216
Node slots: 2
Creating bitmaps: done
Initializing superblock: done
Writing system files: done
Writing superblock: done
Writing backup superblock: 0 block(s)
Formatting Journals: done
Growing extent allocator: done
Formatting slot map: done
Formatting quota files: done
Writing lost+found: done
mkfs.ocfs2 successful

root@HZ-UIS01-CVK01:~#
```

5. 磁盘检查-fsck

```
# fsck [-t 文件系统格式] [-ACay] 磁盘名称
```

选项与参数：

- -t ：指定文件系统类型，不过当前的 Linux 系统自动透过 superblock 去分别文件系统类型，因此通常不需要该参数。
- -A：依据/etc/fstab 的内容，将需要的磁盘扫描一次，通常开机过程就会执行此指令。
- -a：自动修复检查到的有问题的扇区，所以你不用一直按 y 键。
- -y：与-a 类似，但是某些文件系统仅支持-y 这个参数。
- -C：可以在检查的过程中，使用一个直方图来显示目前的进度。

举例：对“/dev/sdb1”磁盘分区进行磁盘检查。

```
root@HZ-UIS01-CVK01:~# fsck -C /dev/sdb1
fsck from util-linux 2.20.1
e2fsck 1.42 (29-Nov-2011)
/dev/sdb1: clean, 11/125184 files, 16807/499744 blocks
```

6. 磁盘挂载-mount

```
# mount [-t 文件系统格式] [-L Lable 名] [-o 额外选项] [-n] 磁盘文件名 挂载点
```

选项与参数：

- -a ：依照配置文件/etc/fstba 的数据将所有未挂载的磁盘都挂上来；

- -l ：单纯的输入 mount 会显示目前挂载的信息，加上-l 可增加列 Lable 名称。
- -t ：可以加上文件系统类型来指定欲挂载的类型。
- -n ：在默认的情况下，系统将会将实际挂载的情况实时写入/etc/mtab 中，以利于其他程序的运行。
- -L ：系统除了利用磁盘文件名外，还可以利用文件系统的标头名称来进程挂载。
- -l ：后面可以接一些挂载时额外加上的参数。比如账号、密码、读取权限等。

举例：挂载“/dev/sdb1”到“/mnt”。

```
root@HZ-UIS01-CVK01:~# mount /dev/sdb1 /mnt
root@HZ-UIS01-CVK01:~# df -Th
Filesystem         Type        Size     Used     Avail    Use%     Mounted on
/dev/sda1            ext4        28G     5.7G      21G      22%     /
udev            devtmpfs        32G     4.0K       32G       1%     /dev
tmpfs                tmpfs        13G    408K       13G       1%     /run
none                 tmpfs       5.0M         0       5.0M       0%     /run/lock
none                 tmpfs        32G     17M       32G       1%     /run/shm
/dev/sda6            ext4       241G     48G     181G      21%    /vms
/dev/sdb1            ext3        1.9G    35M      1.8G       2%      /mnt
```

7. 卸载挂载-umount

```
# umount [-fn]磁盘文件名
```

选项与参数：

- -f ：强制卸载！可以用在类似网络文件系统（NFS）无法读取的情况下；
- -n ：不更新/etc/mtab 情况下卸载

举例：

```
root@HZ-UIS01-CVK01:~# df -Th
Filesystem         Type        Size     Used     Avail    Use%     Mounted on
/dev/sda1            ext4        28G     5.7G      21G      22%     /
udev            devtmpfs        32G     4.0K       32G       1%     /dev
tmpfs                tmpfs        13G    408K       13G       1%     /run
none                 tmpfs       5.0M         0       5.0M       0%     /run/lock
none                 tmpfs        32G     17M       32G       1%     /run/shm
/dev/sda6            ext4       241G     48G     181G      21%    /vms
/dev/sdb1            ext3        1.9G    35M      1.8G       2%      /mnt
root@HZ-UIS01-CVK01:~#
root@HZ-UIS01-CVK01:~# umount /mnt
root@HZ-UIS01-CVK01:~# df -Th
Filesystem         Type        Size     Used     Avail    Use%     Mounted on
/dev/sda1            ext4        28G     5.7G      21G      22%     /
udev            devtmpfs        32G     4.0K       32G       1%     /dev
tmpfs                tmpfs        13G    408K       13G       1%     /run
none                 tmpfs       5.0M         0       5.0M       0%     /run/lock
none                 tmpfs        32G     17M       32G       1%     /run/shm
/dev/sda6            ext4       241G     48G     181G      21%    /vms
```

8. 数据同步写入磁盘-sync

输入命令 sync，那么内存中尚未被更新的数据，就会被写入磁盘中。

举例：

```
root@HZ-UIS01-CVK01:~# sync
root@HZ-UIS01-CVK01:~#
```