第 3 章 Proxmox VE 服务器管理

Proxmox VE 基于著名的 Debian Linux 发行版。也就是说，你可以充分利用 Debian 操作系统的所有软件包资源，以及 Debian 完善的技术文档。可以在线阅读 Debian Administrator's Handbook，深入了解 Debian 操作系统的有关内容（见[Hertzorg13]）。

Proxmox VE 安装后默认配置使用 Debian 的默认软件源，所以你可以通过该软件源直接获取补丁修复和安全升级。此外，我们还提供了 Proxmox VE 自己的软件源，以便升级 Proxmox VE 相关的软件包。这其中还包括了部分必要的 Debian 软件包升级补丁。

我们还为 Proxmox VE 提供了一个专门优化过的 Linux 内核，其中开启了所有必需的虚拟化和容器功能。该内核还提供了 ZFS 相关驱动程序，以及多个硬件驱动程序。例如我们在内核中包含了 Intel 网卡驱动以支持最新的 Intel 硬件设备。

后续章节将集中讨论虚拟化相关内容。有些内容是关于 Debian 和 Proxmox VE 的不同之处，有些是关于 Proxmox VE 的日常管理任务。更多内容请参考 Debian 相关文档。

3.1 软件源

所有基于 Debian 的操作系统都使用 APT 命令作为软件包管理工具。软件源列表定义在/etc/apt/sources.list 文件中，以及/etc/apt/sources.d 目录下后缀名为.list 的文件中。既可以直接使用 apt-get 命令升级软件，也可以使用 GUI 界面升级。

软件源文件 sources.list 的每一行都定义了一个软件源，最常用的软件源一般放在前面。在 sources.list 中，空行会被忽略，字符#及以后的内容会被解析为注释。可以用 apt-get update 命令获取软件源中的软件包信息。

/etc/apt/souces.list 文件

```
deb http://ftp.debian.org/debian jessie main contrib

# security updates

deb http://security.debian.org jessie/updates main contrib.
```

此外，Proxmox VE 还提供了另外三个软件源。

3.1.1 Proxmox VE 企业版软件源

Proxmox VE 企业版软件源是默认的、稳定的、推荐使用的软件源，供订阅了 Proxmox VE 企业版的用户使用。该软件源包含了最稳定的软件包，适用于生产环境使用。软件源 pve-enterprise 默认是启用的。

/etc/apt/sources.list.d/pve-enterprise.list 文件

deb https://enterprise.proxmox.com/debian jessie pve-enterprise

一旦有软件包有新的升级，root@pam 用户就会收到有关新软件包的电子邮件通知。在 GUI 界面，可以查看每个软件包的变更历史（如果有的话），其中有升级的每个细节。所以你永远不会错过重要的安全补丁。

请注意，你必须提供订阅密钥才可以访问企业版软件源。我们提供有不同级别的订阅服务，具体信息可以查看网址 http://www.proxmox.com/en/proxmox-ve/pricing。

➢ 注意

如果你没有订阅 Proxmox VE 企业版，可以将企业版软件源配置信息在软件源配置文件中注释掉（在该行开头插入一个#字符），以避免系统发出错误提示信息。这种情况下可以配置使用 pve-no-subscriptin 软件源。

3.1.2 Proxmox VE 免费版软件源

顾名思义，你无需付费订阅即可访问 Proxmox VE 免费版软件源。该版本适用于测试或非生产环境。我们不推荐在生产环境中使用 Proxmox VE 免费版，因为该版本的软件包往往没有经过足够充分的测试和验证。

我们推荐将免费版软件源配置在/etc/apt/sources.list 文件中。

/etc/apt/sources.list 文件

deb http://ftp.debian.org/debian jessie main contrib

PVE pve-no-subscription repository provided by proxmox.com,

NOT recommended for production use

deb http://download.proxmox.com/debian jessie pve-no-subscription

security updates

deb http://security.debian.org jessie/updates main contrib.

3.1.3 Proxmox VE 测试版软件源

最后，我们还提供了一个名为 pvetest 的测试版软件源。这个版本的软件源包含了最新版本的软件包，主要供开发人员测试新功能和新特性使用。同样，你可以在配置文件 /etc/apt/sources.list 中设置该软件源，如下：

在 **sources.list** 中配置 **pvetest** 的示例

deb http://download.proxmox.com/debian jessie pvetest

☒ 警告

pvetest 软件源仅供新功能测试和 bug 修复补丁测试使用（顾名思义）。

3.1.4 SecureApt

我们使用 GnuPG 为以上软件源中的 Release 文件做了电子签名，APT 通过该电子签名来验证并确保相关软件包来自可信任的软件源。

如果你是用官方 ISO 镜像安装的 Proxmox VE，那么电子签名验证密钥随会一并安装到服务器。如果你使用其他方式安装 Proxmox VE，则可以手动下载验证密钥。下载命令如下：

wget http://download.proxmox.com/debian/key.asc

并可以用如下命令检查密钥

gpg --with-fingerprint key.asc

pub 1024D/9887F95A 2008-10-28 Proxmox Release Key <proxmox-release@pro
xmox .com>

Key fingerprint = BE25 7BAA 5D40 6D01 157D 323E C23A C7F4 9887 F95A

sub 2048g/A87A1B00 2008-10-28

如果密钥检查输出结果和以上完全一致，可以使用如下命令，将该密钥添加为你的可信任 APT 密钥：

```
# apt-key add key.asc
```

3.2 系统软件升级

我们会定期更新 Proxmox VE 三个软件源的软件包。你可以通过 GUI 管理界面安装升级软件包，也可以直接以命令行方式运行 apt-get 升级：

```
apt-get update
apt-get dist-upgrade
```

➢ 注意

apt 软件包管理系统非常灵活，有无数的选项和特性。可以运行 man apt-get 或查看 [Hertzog13]获取相关信息。

你应该定期执行以上升级操作，也可以在我们发布安全更新时执行升级。重大系统升级通知会通过 Proxmox VE Community Forum 发布。随升级通知发布的还会有具体的升级细节。

➢ 提示

我们建议定期升级 Proxmox VE，因为及时获取最新的安全补丁是非常重要的。

3.3 网络配置

Proxmox VE 采用桥接网络模型。每台 Proxmox VE 服务器最多可以设置 4094 个网桥。网桥的功能类似于软件实现的网络交换机。所有的虚拟机可以共享一个网桥，其效果就和连接每台虚拟机的虚拟网线同时插入同一台交换机一样。当然，你也可以配置多个网桥，以便建立互相隔离的网络区域。

为使虚拟机能够连接到外部网络，网桥需要和服务器的物理网卡设备桥接。你也可以进一步利用 VLANs（IEEE 802.1q）和多网口绑定技术，也就是所谓的“链路聚合”，满足更加复杂和多样性的虚拟网络配置需求。

传统上 Debian 使用 ifup 和 ifdown 命令来启停网络接口，所有的网络配置信息都保存在/etc/network/interfaces 中。可以参考相关 man 文档（man interfaces）获取完整的配置方法。

➢ 注意

Proxmox VE 并不直接编辑配置文件/etc/network/interfaces 。实际上，我们会先把网络配置变更保存在临时文件/etc/network/interfaces.new 里，然后在你重启服务器时再用该配置文件覆盖/etc/network/interfaces。

必须提醒的是，你仍然可以直接编辑配置文件来修改网络配置。Proxmox VE 会尽可能跟踪并使用用户设置的最新网络配置。但仍然推荐使用 GUI 来调整网络配置，因为这能让你有效避免意外错误。

3.3.1 命名规范

目前我们采用的网络设备命名规范如下：

- 网卡：eth[N]，其中 0 ≤ N（eth0，eth1， …）
- 网桥：vmbr[N]，其中 0 ≤ N ≤ 4094（vmbr0 - vmbr4094）
- 网口组合：bond[N]，其中 0 ≤ N（bond0，bond1， …）
- VLANs：只需要姜 VLAN 编号附加到网络设备名称后面，并用“.”分隔（eht0.50，bond1.30）

采用命名规范将网络设备名称和网络设备类型关联起来，能够大大降低网络故障排查难度。

3.3.2 基于网桥的默认配置

Proxmox VE 安装程序会创建一个名为 vmbr0 的网桥，并和检测到的服务器第一块网卡 eht0 桥接。配置文件/etc/network/interfaces 中的对应配置信息如下：

```
auto lo
iface lo inet loopback
iface eth0 inet manual
auto vmbr0
iface vmbr0 inet static
        address 192.168.10.2
        netmask 255.255.255.0
        gateway 192.168.10.1
        bridge_ports eth0
        bridge_stp off
        bridge_fd 0
```

在基于网桥的默认配置下，虚拟机看起来就和直接接入物理网络一样。尽管所有虚拟机共享一根网线接入网络，但每台虚拟机都使用自己独立的 MAC 地址访问网络。

3.3.3 路由配置

但大部分 IPC 服务器供应商并不支持基于网桥的默认配置方式，出于网络安全的考虑，一旦发现网络接口上有多个 MAC 地址出现，他们会立刻禁用相关网络端口。

➢ 提示

也有一些 IPC 服务器供应商允许你注册多个 MAC 地址。这就可以避免上面提到的问题，但这样你就需要注册每一个虚拟机 MAC 地址，实际操作会非常麻烦。

你可以用配置“路由”的方式让多个虚拟机共享一个网络端口，这样就可以避免上面提到的问题。这种方式可以确保所有的对外网络通信都使用同一个 MAC 地址。

常见的应用场景是，你有一个可以和外部网络通信的 IP 地址（假定为 192.168.10.2），还有一个供虚拟机使用的 IP 地址段（10.10.10.1/255.255.255.0）。针对该场景，我们推荐使用如下配置：

```
auto lo
iface lo inet loopback
```

```
auto eth0
iface eth0 inet static
        address 192.168.10.2
        netmask 255.255.255.0
        gateway 192.168.10.1
        post-up echo 1 > /proc/sys/net/ipv4/conf/eth0/proxy_arp

auto vmbr0
iface vmbr0 inet static
        address 10.10.10.1
        netmask 255.255.255.0
        bridge_ports none
        bridge_stp off
        bridge_fd 0
```

3.3.4 基于 iptables 的网络地址转换配置（NAT）

某些情况下，你可能希望采用 NAT 技术，以在使用 Proxmox 服务器真实地址替代虚拟机的私有地址和外部网络通信。参考配置如下：

```
auto lo
iface lo inet loopback

auto eth0
#real IP address
iface eth0 inet static
        address 192.168.10.2
        netmask 255.255.255.0
        gateway 192.168.10.1
```

```
auto vmbr0
#private sub network
iface vmbr0 inet static
        address 10.10.10.1
        netmask 255.255.255.0
        bridge_ports none
        bridge_stp off
        bridge_fd 0
        post-up echo 1 > /proc/sys/net/ipv4/ip_forward
        post-up iptables -t nat -A POSTROUTING -s ' 10.10.10.0/24' -o eth0
                -j MASQUERADE
        post-down iptables -t nat -D POSTROUTING -s ' 10.10.10.0/24' -o eth0
                -j MASQUERADE
```

3.3.5 Linux 多网口绑定

多网口绑定（也称为网卡组或链路聚合）是一种将多个网卡绑定成单个网络设备的技术。利用该技术可以实现某个或多个目标，例如提高网络链路容错能力，增加网络通信性能等。

类似光纤通道和光纤交换机这样的高速网络硬件的价格一般都非常昂贵。利用链路聚合技术，将两个物理网卡组成一个逻辑网卡，能够将网络传输速度加倍。大部分交换机设备都已经支持 Linux 内核的这个特性。如果你的服务器有多个以太网口，你可以将这些网口连接到不同的交换机，以便将故障点分散到不同的网络设备，一旦有物理线路故障或网络设备故障发生，多网卡绑定会自动将通信流量从故障线路切换到正常线路。

链路聚合技术可以有效减少虚拟机在线迁移的时延，并提高 Proxmox VE 集群服务器节点之间的数据复制速度。

目前一共有 7 种网口绑定模式：

- 轮询模式（blance-rr）：网络数据包将按顺序从绑定的第一个网卡到最后一个网卡轮流发送。这种模式可以同时实现负载均衡和链路容错效果。

- 主备模式（active-backup）：该模式下网卡组中只有一个网卡活动。只有当活动的网卡故障时，其他网卡才会启动并接替该网卡的工作。整个网卡组使用其中一块网卡的 MAC 地址作为对外通信的 MAC 地址，以避免网络交换机产生混乱。这种模式仅能实现链路容错效果。

- 异或模式（balance-xor）：网络数据包按照异或策略在网卡组中选择一个网卡发送（[源 MAC 地址 XOR 目标 MAC 地址] MOD 网卡组中网卡数量）。对于同一个目标 MAC 地址，该模式每次都选择使用相同网卡通信。该模式能同时实现负载均衡和链路容错效果。

- 广播模式（broadcast）：网络数据包会同时通过网卡组中所有网卡发送。该模式能实现链路容错效果。

- IEEE 802.3ad 动态链路聚合模式（802.3ad）（LACP）：该模式会创建多个速度和双工配置一致的聚合组。并根据 802.3ad 标准在活动聚合组中使用所有网卡进行通信。

- 自适应传输负载均衡模式（balance-tlb）：该 Linux 网卡绑定模式无须交换机支持即可配置使用。根据当前每块网卡的负载情况（根据链路速度计算的相对值），流出的网络数据包流量会自动进行均衡。流入的网络流量将由当前指定的一块网卡接收。如果接收流入流量的网卡故障，会自动重新指定一块网卡接收网络数据包，但该网卡仍沿用之前故障网卡的 MAC 地址。

- 自适应负载均衡模式（均衡的 IEEE 802.3ad 动态链路聚合模式（802.3ad）（LACP）:-alb）：该模式是在 blance-tlb 模式的基础上结合了 IPV4 网络流量接收负载均衡（rlb）特性，并且无须网络交换机的专门支持即可配置使用。网络流量接收负载均衡基于 ARP 协商实现。网卡组驱动将自动截获本机的 ARP 应答报文，并使用网卡组中其中一块网卡的 MAC 地址覆盖 ARP 报文中应答的源 MAC 地址，从而达到不同的网络通信对端和本机不同 MAC 地址通信的效果。

大多数情况下，active-backup 模式会是最佳选择。但如果你的网络交换机支持 LACP “IEEE 802.3ad”功能，那么 802.3ad 模式会是更好的选择。

下面所列的网卡绑定配置示例可用于分布式/共享存储网络配置。其主要优势是能达到更高的传输速度，同时实现网络链路容错的效果。

示例：基于固定 IP 地址的多网卡绑定

```
auto lo
iface lo inet loopback
```

```
iface eth1 inet manual

iface eth2 inet manual

auto bond0
iface bond0 inet static
        slaves eth1 eth2
        address 192.168.1.2
        netmask 255.255.255.0
        bond_miimon 100
        bond_mode 802.3ad
        bond_xmit_hash_policy layer2+3

auto vmbr0
iface vmbr0 inet static
        address 10.10.10.2
        netmask 255.255.255.0
        gateway 10.10.10.1
        bridge_ports eth0
        bridge_stp off
        bridge_fd 0
```

另一种配置方法是直接使用网卡组作为虚拟交换机桥接端口，从而实现虚拟机网络的容错效果。

```
auto lo
iface lo inet loopback

iface eth1 inet manual

iface eth2 inet manual
```

```
auto bond0
iface bond0 inet maunal
        slaves eth1 eth2
        bond_miimon 100
        bond_mode 802.3ad
        bond_xmit_hash_policy layer2+3

auto vmbr0
iface vmbr0 inet static
        address 10.10.10.2
        netmask 255.255.255.0
        gateway 10.10.10.1
        bridge_ports bond0
        bridge_stp off
        bridge_fd 0
```

3.4 时钟同步

Proxmox VE 集群的正常运行基于一个重要条件，那就是所有节点的服务器时钟需要精确地同步。其他一些组件，例如 Ceph，如果各节点的时钟不同步，也不能正常工作。

各节点之间的时钟同步可利用“网络时间协议”（NTP）实现。Proxmox VE 默认使用 systemd-timesyncd 作为 NTP 客户端，并默认配置使用一组互联网服务器作为时间源。大部分情况下，系统安装后即可自动实现时钟同步。

3.4.1 使用自定义 NPT 服务器

大多数情况下，默认的 NTP 服务器可能并不适用。例如你的 Proxmox VE 服务器根本连不上 internet 互联网（比如，防火墙策略禁止了互联网连接），这时你需要自行搭建本地 NTP 服务器，并配置 systemd-timesyncd 和本地 NTP 服务器进行时钟同步。

/etc/systemd/timesyncd.conf 默认配置

[Time]

Servers=ntp1.example.com ntp2.example.com ntp3.example.com ntp4.example.com

重启时钟同步服务后（systemctl restart systemd-timesyncd），你可以查看日志以验证新配置的 NTP 服务器是否已被启用（journalctl –since -1h -u systemd-timesyncd）。

日志输出示例如下：

…

Oct 07 14:58:36 node1 systemd[1]: Stopping Network Time Synchronization...

Oct 07 14:58:36 node1 systemd[1]: Starting Network Time Synchronization...

Oct 07 14:58:36 node1 systemd[1]: Started Network Time Synchronization.

Oct 07 14:58:36 node1 systemd-timesyncd[13514]: Using NTP server 10.0.0.1:123 (ntp1.example.com).

Oct 07 14:58:36 nora systemd-timesyncd[13514]: interval/delta/delay/jitter/drift 64s/-0.002s/0.020s/0.000s/-31ppm

…

3.5 外部监控服务器

从 Proxmox VE 4.0 开始，你可以搭建外部监控服务器，集中收集 Proxmox VE 服务器主机、虚拟机和存储的监控数据。

目前支持的外部监控服务器有：

- graphite（参见 http://graphiteapp.org ）
- influxdb（参见 https://www.influxdata.com/time-series-platform/influxdb ）

外部监控服务器配置信息保存在/etc/pve/status.cfg 文件中。

3.5.1 配置 Graphite 服务器

Graphite 服务器可按如下格式配置

graphite:

server your-server

port your-port

path your-path

其中 your-port 默认设置为 2003，your-path 默认设置为 proxmox。

Proxmox VE 服务器使用 udp 协议发送监控数据，所以 graphite 服务器需要进行相应配置。

3.5.2 配置 Influxdb 插件

可按如下格式配置

influxdb:

server your-server

port your-port

Proxmox VE 服务器使用 udp 协议发送监控数据，所以 influxdb 服务器需要进行相应配置。

下面是一个 influxdb 配置示例（配置在 influxdb 服务器上）：

[[udp]]

enabled = true

bind-address = "0.0.0.0:8089"

database = "proxmox"

batch-size = 1000

batch-timeout = "1s"

按如上配置，influxdb 服务器将监听 8089 端口，并将接收到的数据写入 proxmox 数据库。

3.6 硬盘健康状态监控

为 Proxmox VE 配置存储设备时，不仅要考虑健壮性和冗余性，还应该考虑监控硬盘的健康状态。

从 4.3 版开始，Proxmox VE 集成了 smartmontools 软件包。这是一套可用来监控管理服务器本地硬盘 S.M.A.R.T 系统的工具。

你可以运行如下命令获取磁盘的状态信息：

```
# smartctl -a /dev/sdX
```

其中/dev/sdX 是某个本地磁盘的设备文件名称。

如果命令输出如下结果：

```
SMART support is: Disabled
```

你就需要运行以下命令启用 SMART 监控：

```
# smartctl -s on /dev/sdX
```

你可以运行 man smartctl 命令查看技术手册，以获取关于 smartctl 使用方法的更多信息。

默认状态下，smartctltools 守护进程 smartd 将会被启动并自动运行，每 30 分钟对/dev/sdX 和/dev/hdX 进行一次扫描，如果检测到的错误和警告信息，就向服务器的 root 用户发送 e-mail。

可以运行 man smartd 和 man smartd.conf 命令，以获取关于 smartd 配置方法的更多信息。

如果你的服务器使用了硬件 raid 卡管理硬盘，一般都会有专门工具来监控 raid 阵列中的硬盘以及 raid 阵列本身。具体可以咨询 raid 卡控制器的厂商。

3.7 逻辑卷管理器（LVM）

大部分用户都直接将 Proxmox VE 安装到服务器本地硬盘。Proxmox VE 安装光盘为服务器本地硬盘配置提供了多种选项，目前默认的选项为 LVM。默认情况下，安装程序会引导你选择一块硬盘用于安装 Proxmox VE 系统，并将该硬盘格式化为物理卷（PV）的形式以创建卷组（VG）pve。下面是在一个 8GB 容量的小硬盘上测试安装 Proxmox VE 后的 LVM 配置输出：

pvs

PV VG Fmt Attr PSize PFree

/dev/sda3 pve lvm2 a-- 7.87g 876.00m

vgs

VG #PV #LV #SN Attr VSize VFree

pve 1 3 0 wz--n- 7.87g 876.00m

安装程序会在 VG 上创建三个逻辑卷（LV）

lvs

LV VG Attr LSize Pool Origin Data% Meta%

data pve twi-a-tz-- 4.38g 0.00 0.63

root pve -wi-ao---- 1.75g

swap pve -wi-ao---- 896.00m

- **root**

格式化为 ext4 文件系统，用于安装 Proxmox VE 操作系统

- **swap**

用于 Swap 分区

- **data**

使用 LVM-thin 格式化，用于保存虚拟机镜像。LVM-thin 在对于虚拟机快照和克隆的效率较高，所以非常适合用于保存虚拟机镜像。

到 Proxmox VE 4.1 为止，安装程序会创建一个标准的名为“data”的逻辑卷，并挂载在/var/lib/vz 路径下。

从 Proxmox VE 4.2 开始，安装程序会以 LVM-thin 形式创建“data”逻辑卷，并用于存储基于块存储的虚拟机镜像，而/var/lib/vz 是根文件系统下的一个路径。

3.7.1 硬件

我们强烈推荐使用基于硬件 RAID 控制器（带有电池保护写缓存，BBU）的服务器安装 Proxmox VE。这样不仅能提高性能，提供冗余性，还能简化故障硬盘的更换操作（，可热插拔）。

LVM 本身无须任何特殊的硬件支持，并且对于内存的需求也很低。

3.7.2 启动引导程序

默认情况下，安装程序会安装两个启动引导程序。第一个分区会安装标准的 GRUB 启动引导程序。第二个分区会格式化为 EFI 系统分区（EFI System Partition，ESP），这样就可以启动基于 EFI 的系统。

3.7.3 创建卷组

假定我们现在有一块空白磁盘设备，其设备文件为/dev/sdb，可以用如下方法利用该磁盘创建卷组“vmdata”。

☒ 警告

执行以下这些命令将会彻底销毁/dev/sdb 上的原有数据。

首先创建一个分区。

```
# sgdisk -N 1 /dev/sdb
```

然后使用静默方式创建一个物理卷（PV），并设置元数据大小为 250K。

```
# pvcreate --metadatasize 250k -y -ff /dev/sdb1
```

然后在/dev/sdb1 上创建卷组“vmdata”。

vgcreate vmdata /dev/sdb1

3.7.4 为/var/lib/vz 添加 LV

可用如下命令添加“薄模式”的逻辑卷（thin LV）。

lvcreate -n <Name> -V <Size[M,G,T]> <VG>/<LVThin_pool>

示例如下：

lvcreate -n vz -V 10G pve/data

然后在该逻辑卷上创建文件系统，如下。

mkfs.ext4 /dev/data/vz

最后，将该逻辑卷挂载到文件系统。

☒ 警告

挂载前需确保/var/lib/vz 下为空。默认/var/lib/vz 不为空。

为确保系统重启后自动挂载，可运行如下命令，修改/etc/fstab 文件的配置内容。

echo ’/dev/pve/vz /var/lib/vz ext4 defaults 0 2’ >> /etc/fstab

3.7.5 调整 thin pool 的容量

可以用如下命令调整 LV 容量大小和元数据池大小。

lvresize --size +<size[\M,G,T]> --poolmetadatasize +<size[\M,G]> <
VG>/<LVThin_pool>

➢ 注意

扩展数据池容量的同时，必须相应扩展元数据池的容量。

3.7.6 创建 LVM-thin 存储池

首先要创建一个卷组，然后才能创建“薄模式”存储池。可以参考 LVM 一节查看创建卷组的步骤。创建“薄模式”存储池命令如下。

```
# lvcreate -L 80G -T -n vmstore vmdata
```

3.8 Linux 上的 ZFS

ZFS 是由太阳微系统公司（Sun Microsystems）设计的同时集成有文件管理功能和逻辑卷管理功能的文件系统。从 Proxmox VE 3.4 开始，Proxmox VE 引入了 ZFS 文件系统作为可选的文件系统和根文件系统。Proxmox VE 官方 ISO 光盘镜像已经集成了 ZFS 所需的软件包，用户无须手工编译即可直接使用 ZFS。

利用 ZFS，用户可以在配置有限的硬件设备上尽可能多的享受企业级的文件系统服务，进一步还可以利用 SSD 缓存或者全 SSD 技术获得极高的系统性能。利用常见的 CPU 和内存硬件，只需简单的配置，ZFS 即可实现昂贵的基于硬件 RAID 卡的管理功能。

ZFS 的技术优势如下：

- 可通过 Proxmox VE 的图形界面或命令行方式进行配置管理，容易使用。
- 高可靠性
- 有效防止数据损坏丢失
- 文件系统级的数据压缩
- 支持快照功能
- 支持 Copy-on-write
- 多种软 RAID 模式：RAID0，RAID1，RAID10，RAIDZ-1，RAIDZ-2，RAIDZ-3
- 支持 SSD 缓存
- 自我数据修复
- 连续数据完整性检查和验证

- 支持大容量数据存储
- 有效防止数据损坏丢失
- 基于网络的数据异步复制
- 开源软件
- 支持数据加密

3.8.1 硬件

ZFS 对于内存配置的依赖较高，一般最少需要为 ZFS 配置 8GB 内存。实际生产中，最好基于你的预算配置尽可能多的内存。为防止数据损坏，我们建议你使用高端的 ECC 内存条。

如果你要为 ZFS 配置独立的缓存盘或文件系统日志盘，最好使用企业级的 SSD 盘（例如 Intel SSD DC S3700 系列）。这能够极大的提升整体性能表现。

☒ 重要

不要使用有独立缓存管理的硬件控制器。ZFS 需要能够直接访问磁盘。ZFS 可以和 HBA 卡配合使用，也可以和“IT”模式的 LSI 控制器及类似硬件配合使用。

如果你是在 Proxmox VE 虚拟机环境里测试安装 Proxmox VE（嵌套虚拟化），不要使用 virtio 类型的虚拟磁盘，目前 ZFS 不支持 virtio 类型磁盘。建议为虚拟机配置 IDE 或 SCSI 类型的虚拟磁盘（virtio SCSI 控制器）。

3.8.2 用于根文件系统

当你使用 Proxmox VE 安装程序安装时，可以选择使用 ZFS 作为根文件系统。同时你需要在安装过程中选择配置 RAID 级别。

- **RAID0**

也称为“条带模式”。该模式下 ZFS 卷的容量为所有硬盘容量之和。但 RAID0 不提供任何冗余性，卷中任何一块硬盘故障都会导致整个卷不可用。

- **RAID1**

也称为“镜像模式”。该模式下，数据会以复制方式同时写入所有硬盘。该模式至少需要 2 块容量一样的硬盘，而整个卷的容量就等于单块盘的容量。

- **RAID10**

该模式组合了 RAID0 和 RAID1 模式。配置使用该模式至少需要 4 块硬盘。

- **RAIDZ-1**

类似于 RAID5 模式，提供 1 块硬盘故障冗余。配置使用该模式至少需要 3 块硬盘。

- **RAIDZ-2**

类似于 RAID5 模式，提供 2 块硬盘故障冗余。配置使用该模式至少需要 4 块硬盘。

- **RAIDZ-3**

类似于 RAID5 模式，提供 3 块硬盘故障冗余。配置使用该模式至少需要 5 块硬盘。

安装程序会自动完成硬盘分区，构建名为“rpool”的 ZFS 存储池，并在 rpool/ROOT/pve-1 上安装根文件系统。

安装程序还会创建一个名为 rpool/data 的子卷，用于保存虚拟机镜像。为便于使用 Proxmox VE 工具管理该 ZFS 卷，安装程序会在/etc/pve/storage.cfg 文件中创建以下配置信息：

```
zfspool: local-zfs
        pool rpool/data
        sparse
        content images,rootdir
```

安装完成后，你可以运行 zpool 命令查看 ZFS 存储池的状态：

```
# zpool status
pool: rpool
state: ONLINE
scan: none requested
config:
        NAME            STATE           READ   WRITE CKSUM
```

rpool	ONLINE	0	0	0
mirror-0	ONLINE	0	0	0
sda2	ONLINE	0	0	0
sdb2	ONLINE	0	0	0
mirror-1	ONLINE	0	0	0
sdc	ONLINE	0	0	0
sdd	ONLINE	0	0	0

errors: No known data errors

可以用 ZFS 命令配置管理 ZFS 文件系统。下面的命令用于列出安装 Proxmox VE 后的 ZFS 文件系统。

zfs list

NAME	USED	AVAIL	REFER	MOUNTPOINT
rpool	4.94G	7.68T	96K	/rpool
rpool/ROOT	702M	7.68T	96K	/rpool/ROOT
rpool/ROOT/pve-1	702M	7.68T	702M	/
rpool/data	96K	7.68T	96K	/rpool/data
rpool/swap	4.25G	7.69T	64K	-

3.8.3 系统启动程序

默认情况下，ZFS 分区会保留硬盘的前 2048 个扇区，足够 GRUB 启动分区安装使用。Proxmox VE 安装程序会自动将 GRUB 启动程序安装到该空间。如果你配置的 RAID 级别提供了故障冗余，安装程序会把启动程序复制到系统 ZFS 卷的所有硬盘上，这样即使部分磁盘故障，系统仍然能够顺利启动。

➢ 注意

ZFS 根文件系统不支持使用 UEFI 系统启动程序引导启动。

3.8.4 ZFS 管理

本节将给出 ZFS 日常管理操作的范例。ZFS 本身非常强大，具有很多命令选项。管理 ZFS 最主要的两个命令是 zfs 和 zpool。这两个命令都有非常完善的技术手册，可以使用如下命令查看：

man zpool

man zfs

- 创建新的存储池

至少需要有 1 块硬盘才可以创建一个新的存储池。可以用参数 ashift 指定区块大小（2 的 asshift 次幂），且不能小于所用磁盘区块的大小。

zpool create -f -o ashift=12 <pool> <device>

还可以执行以下命令激活数据压缩功能。

zfs set compression=lz4 <pool>

- 创建新的 **RAID-0** 存储池

至少需要 1 块硬盘。

zpool create -f -o ashift=12 <pool> <device1> <device2>

- 创建新的 **RAID-1** 存储池

至少需要 2 块硬盘。

zpool create -f -o ashift=12 <pool> mirror <device1> <device2>

- 创建新的 **RAID-10** 存储池

至少需要 4 块硬盘。

zpool create -f -o ashift=12 <pool> mirror <device1> <device2> mirror <device3> <device4>

- 创建新的 **RAIDZ-1** 存储池

至少需要 3 块硬盘。

zpool create -f -o ashift=12 <pool> raidz1 <device1> <device2> <device3>

- 创建新的 **RAIDZ-2** 存储池

至少需要 4 块硬盘。

zpool create -f -o ashift=12 <pool> raidz2 <device1> <device2> <device3> <device4>

- 创建新的带有缓存（**L2ARC**）的存储池

可以使用独立的硬盘分区（建议使用 SSD）作为缓存，以提高 ZFS 性能。如下命令中，<device>处可以列出多个硬盘设备，命令格式就像“创建带 RAID 的存储池”一样。

zpool create -f -o ashift=12 <pool> <device> cache <cache_device>

- 创建新的带日志（**ZIL**）的存储池

可以使用独立的硬盘分区（建议使用 SSD）记录文件系统日志，以提高 ZFS 性能。如下命令中，<device>处可以列出多个硬盘设备，命令格式就像“创建带 RAID 的存储池”一样。

zpool create -f -o ashift=12 <pool> <device> log <log_device>

- 为已有的存储池添加缓存和日志盘

如果你要为一个未配置缓存和日志盘的 ZFS 存储池添加缓存和日志盘，首先需要使用 parted 或者 gdisk 将 SSD 盘划分为两个分区。

☒ 重要

确保使用 GPT 分区表。

日志盘的大小最大为物理内存容量的一半，通常都不大。SSD 盘剩余空间可用作缓存。

zpool add -f <pool> log <device-part1> cache <device-part2>

- 更换故障磁盘

zpool replace -f <pool> <old device> <new-device>

3.8.5 使用邮件通知

ZFS 有一个事件守护进程，专门监控 ZFS 内核模块产生的各类事件。当发生严重错误，例如存储池错误时，该进程还可以发送邮件通知。

可以编辑配置文件/etc/zfs/zed.d/zed.rc 以激活邮件通知功能。只需将配置参数 ZED_EMAIL_ADDR 前的注释符号去除即可，如下：

ZED_EMAIL_ADDR="root"

请注意，Proxmox VE 会将邮件发送给为 root 用户配置的电子邮件地址。

☒ 重要

只需激活 ZED_MAIL_ADDR 参数即可。其他配置参数均为可选项目，非必须项。

3.8.6 配置 ZFS 内存使用上限

最好将 ZFS 的缓存容量 ZFS ARC 上限设置为物理内存容量的一半，以避免内存短缺导致系统性能变坏。具体做法是编辑配置文件/etc/modprobe.d/zfs.conf，插入如下内容：

options zfs zfs_arc_max=8589934592

上例中设置 ZFS 缓存容量上限为 8GB。

☒ 重要

如果根文件系统也使用了 ZFS，你必须在每次修改该参数后更新 initramfs，如下：

update-initramfs –u

- **ZFS 上的 SWAP**

在 Linux 上将 ZFS 用做 SWAP 分区可能会导致一些问题，例如系统卡死或者很高的 IO 负载。特别是在向外部存储备份文件时会容易触发此类问题。

我们强烈建议为 ZFS 配置足够的物理内存，避免系统出现可用内存不足的情形。此外，你可以调低“swappiness”参数值。通常，设置为 10 比较好。

sysctl -w vm.swappiness=10

如果需要将 swappiness 参数设置持久化，可以编辑文件/etc/sysctl.conf，插入下内容：

vm.swappiness=10

表 3.1 Linux 内核 swappiness 参数设置表

值	对应策略
vm.swappiness=0	内核仅在内存耗尽时进行 swap。
vm.swappiness=1	内核仅执行最低限度的 swap。
vm.swappiness=10	当系统有足够多内存时，可考虑使用该值，以提高系统性能。
vm.swappiness=60	默认设置值。
vm.swappiness=100	内核将尽可能使用 swap。