架構當代一體化儲存與虛擬化與資料救援思路

Build all in one Storage & Virtualization Server

自我簡介

OSSLab thx(熊大)

資歷

- 2013 2015 台灣駭客年會講師
- 台灣警政署資安教育訓練講師
- OSSLab 開放軟體實驗室創辦人
- 專長 撿便宜貨 惡搞

OSSLab 開放軟體實驗室

- 創立於2007年
- 專精於 Voip,儲存 Hack Hardware 領域.
- www.osslab.org.tw
- www.osslab.com.tw

一體儲存與虛擬化

現在電腦硬體非常便宜 資源 **(CPU ,DRAM ,Storage)**使用不完

能將虛擬化與儲存整合在一起 能最大利用資源

利用本機儲存裝置 效能遠大於昂貴**Cluster Storage**

市面上商業產品

Qnap TDS Server
強調 **Application computing server**（應用運算伺服器）與 **data storage server**（資料儲存伺服器）

儲存服務有哪些

File level網路儲存
SMB ,NFS,AFP .

Block Level
iSCSI ,FC ,虛擬磁帶 都是Block level等級網路儲存

Block level target實現方法

Linux SCST
Generic SCSI Target Subsystem for Linux
https://sourceforge.net/p/scst/

Solaris Comstar
Common Multiprotocol SCSI Target

功能
Port providers :Implement protocols .iSCSI .FC .SAS 等.

Logical unit providers 模擬與切分scsi Lun.

目標**Lan ,FC ,infiniband ,SAS** 通道上模擬虛擬硬碟

SDS (Software Define storage)

傳統Storage主控是客製化綁死的客製化ASIC 晶片.
比如說 SAN Storage 主控板.

SDS是指在開放硬體平台上(如x86 ,Arms),先載入Linux.再啟用各種儲存服務,模擬Lun則由上頁二個軟體達成

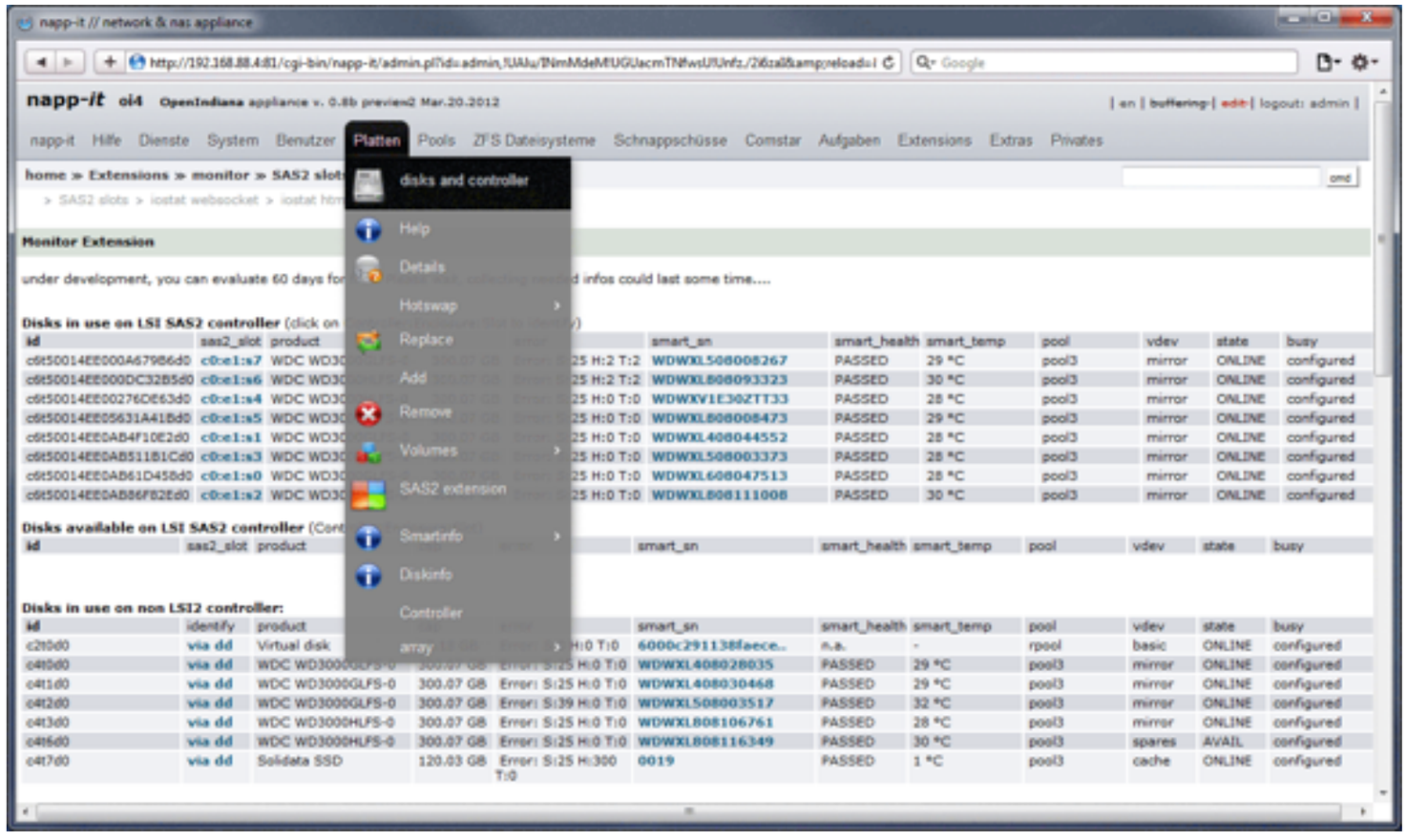
有名的SDS
1.Open-E
2.Nextena
3.Openfiler
4.Solaris 體系OS
5.Vsan
6.NetApp ONTAP Select

Napp it Solaris SDS應用的管理介面

Solaris 體系的OS

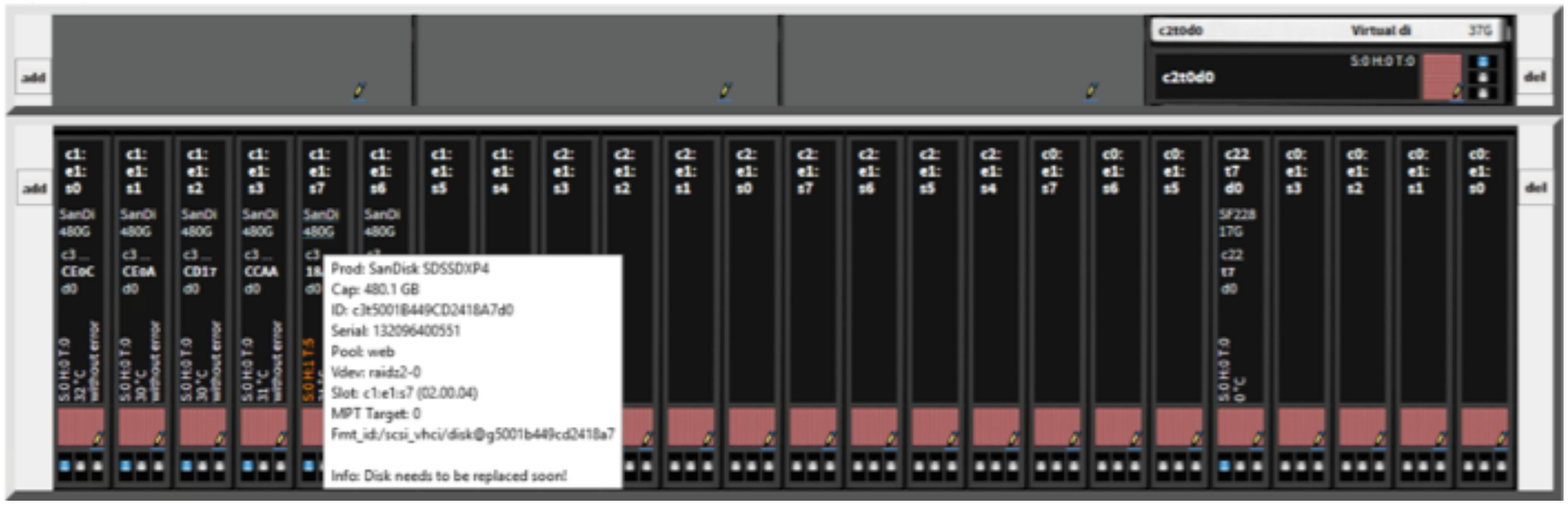
Illumos
Openpedia
OmniOS
nexenta

這些都有管理不易的困擾
Nappit 為此儲存管理外掛
GUI



SDS 管理需要做到怎樣的事情

1.設定
　ACL ,建立 Lun ,HA 等設定.
2.調校
　內建測速程式
3.監控
Dtreace



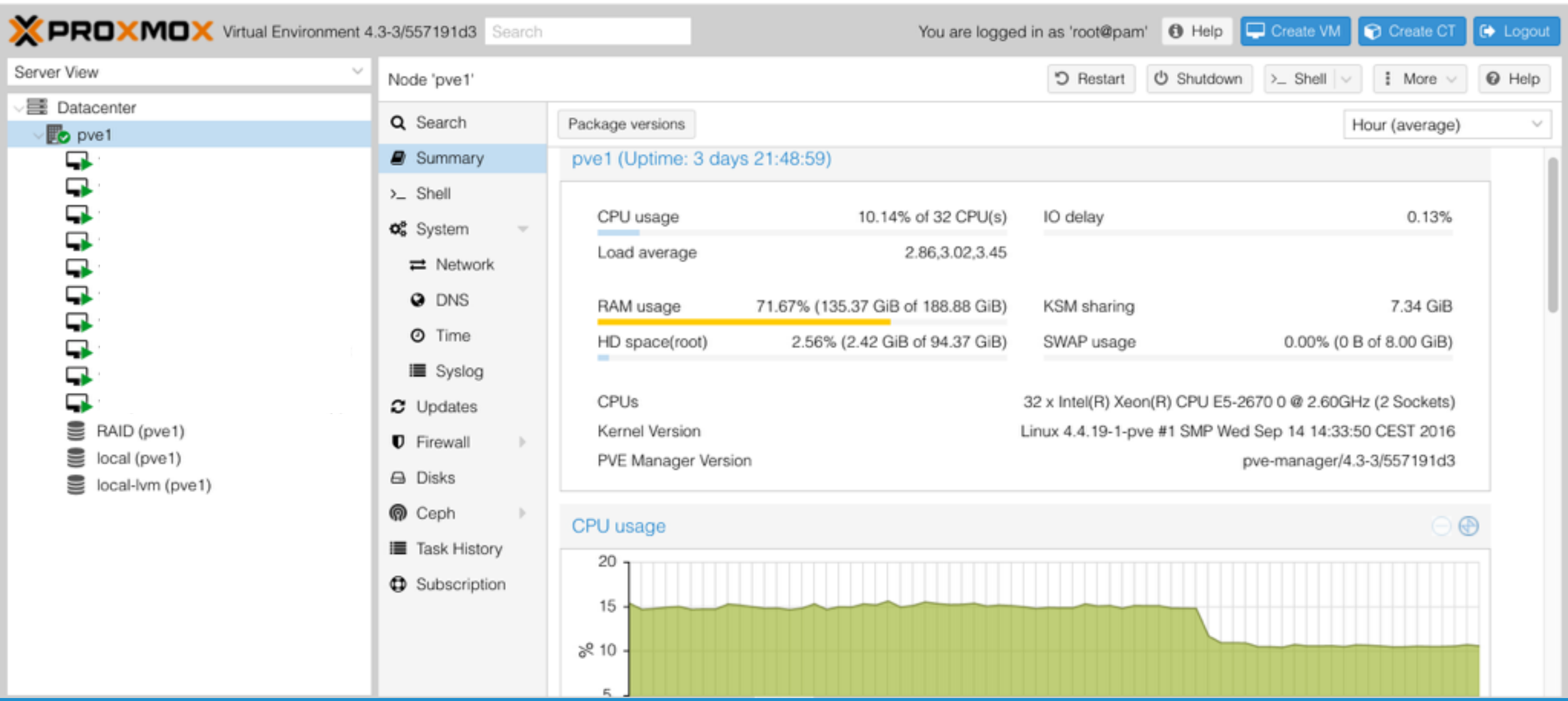
napp-it 通過軟體可監控實際上的硬碟被如何耗用，支援透過**SATA** 或 **LSI HBAs** 所連接並使用 **sasircu** 管控的硬碟，這畫面可以自行畫出。

Proxmox ve 虛擬化平台

1.提供以 **KVM** 虛擬機器技術為基礎的管理平台。
2.提供以 **Container-based** 為基礎的 **OpenVZ** 虛擬化技術
3.可執行不停機的 **Live Migration** 虛擬機器線上移轉。
4提供高可用性的叢集技術，讓運行在 **HA** 架構下的虛擬機器在單一節點因故停擺時，自動重新啟動在另一個可用節點上繼續運行。
5.簡潔有力的**Web** 管理介面來集中控管整個**Hypervisor**。

有這樣多的功能 還免費

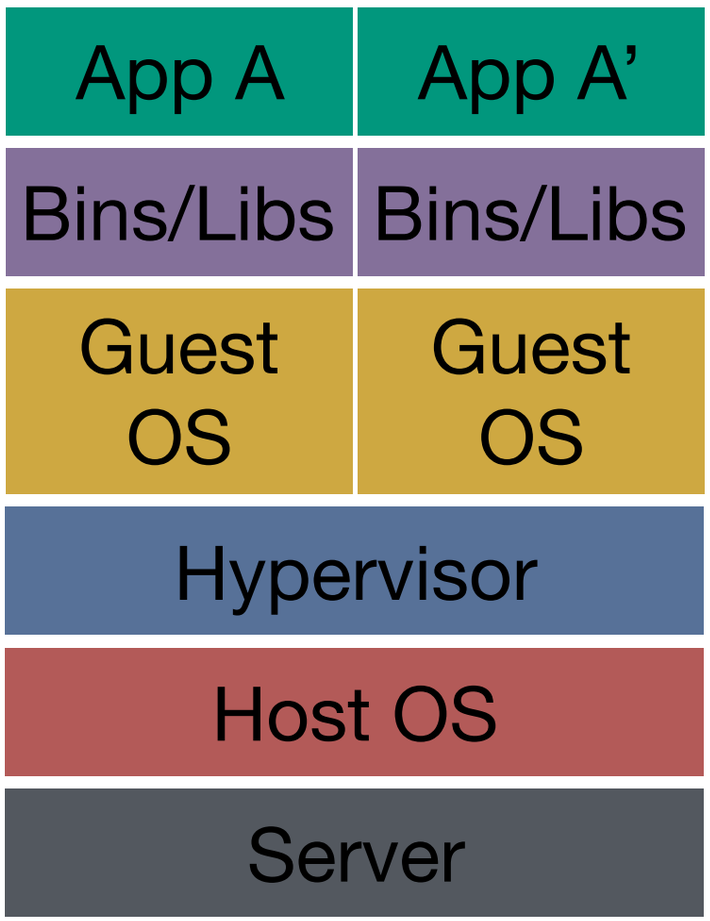
Proxmox ve 虛擬化介面



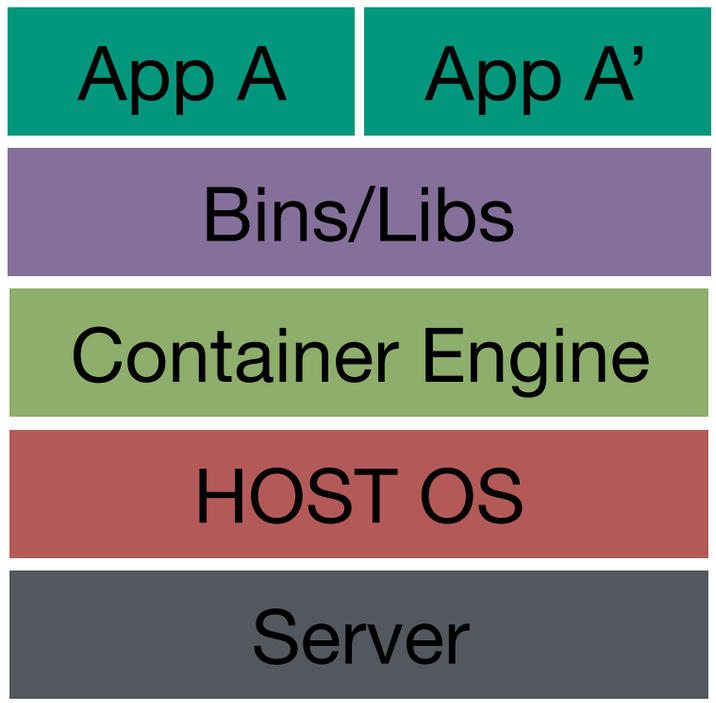
Proxmox建構起來的主機可用資源

Proxmox VE同時支持VM跟容器

VM



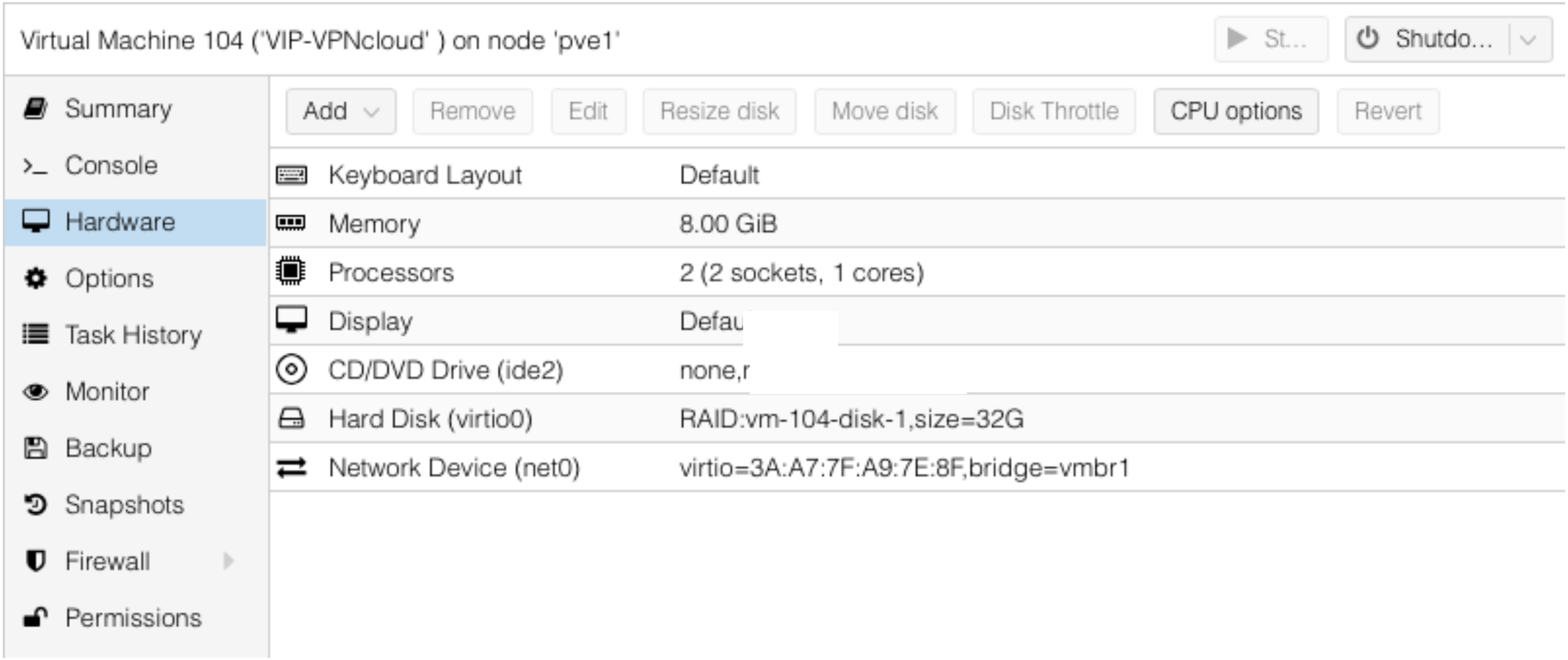
Container



Container 是一種比較新的應用，不需要透過 Hypervisor 來模擬所有的硬體裝置，而是直接可以使用實體主機硬體裝置的虛擬系統架構，讓虛擬化的電腦能達到最大的效能。Linux Container 就是可以直接使用實體電腦上的軟硬體資源，而最特別的地方，就是 Linux Container 與實體系統共享相同的核心與函式庫。

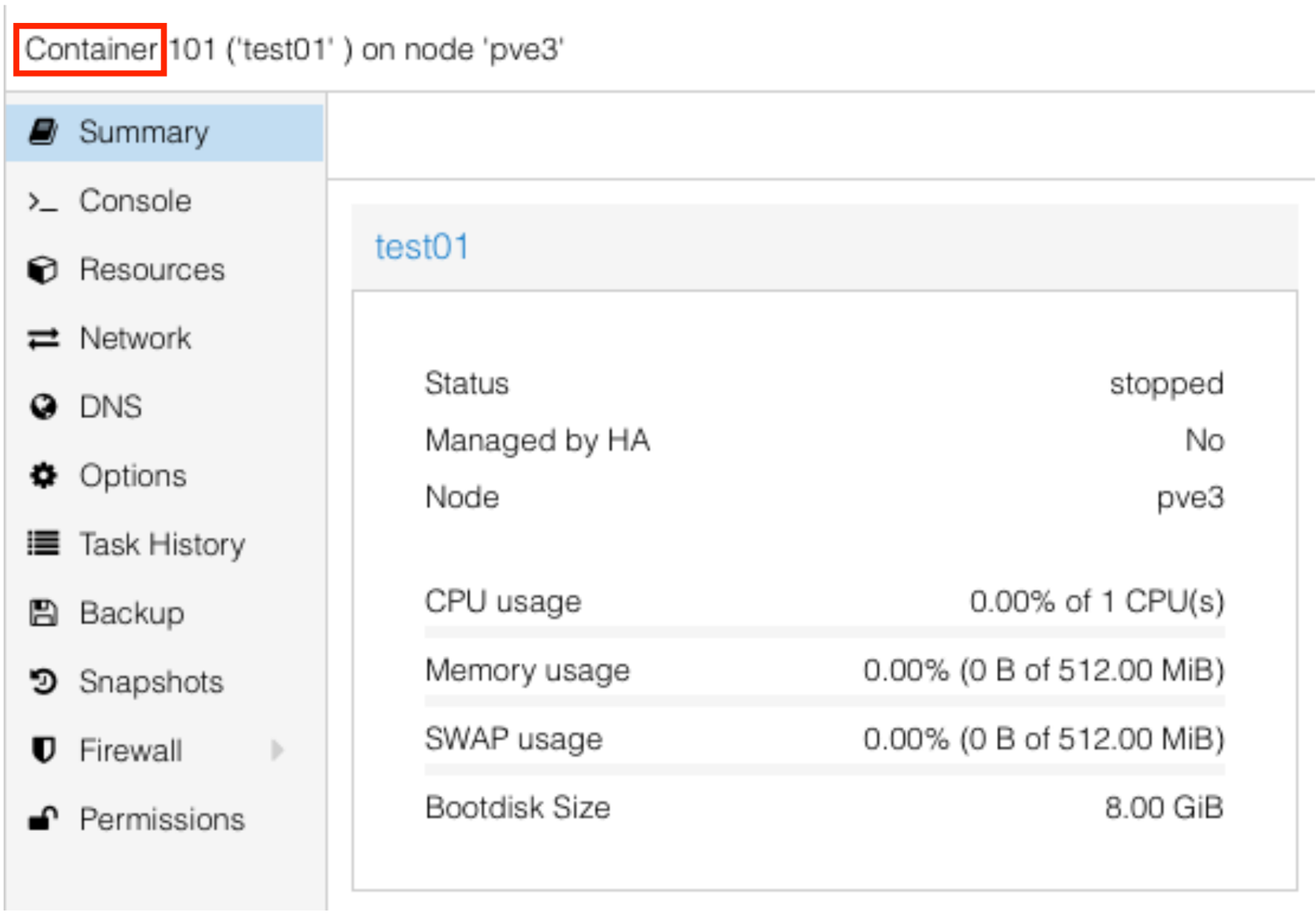
KVM虛擬機

KVM VM 配置顯示.



Proxmox VE 容器標示

左上角會顯示這是 Container，VM 的話會顯示 Virtual Machine



Proxmox內建的容器範本

在這邊選擇所需要的容器範本進行下載，之後才有容器可以使用

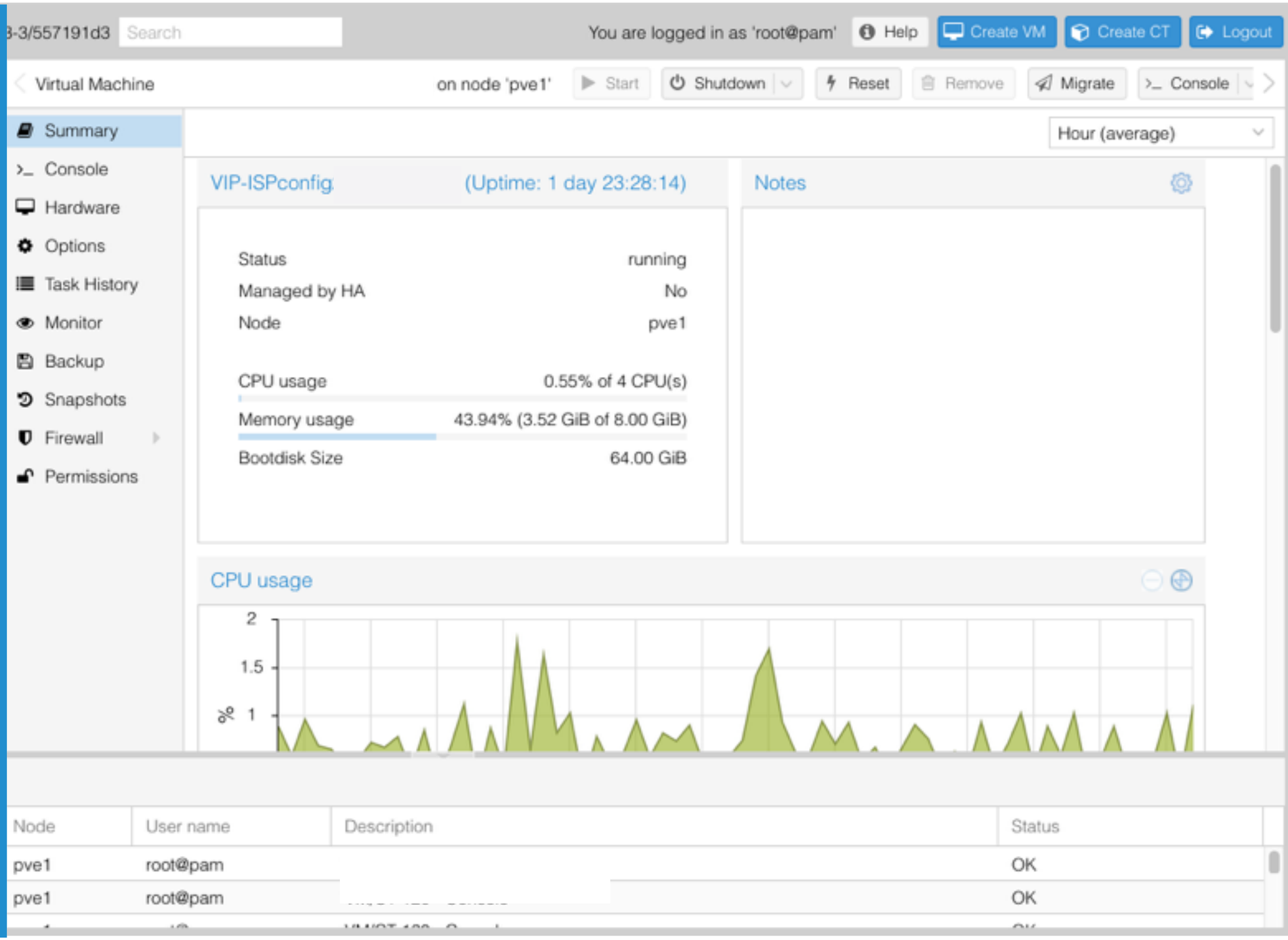
Templates

Type	Package	Version	Description
Section: system (13 Items)			
lxc	ubuntu-15.04-standard	15.04-1	Ubuntu Vivid (standard)
lxc	debian-6.0-standard	6.0-7	Debian 6.0 (standard)
lxc	alpine-3.3-default	20160427	LXC default image for alpine 3.3 (20160427)
lxc	debian-7.0-standard	7.0-3	Debian 7.0 (standard)
lxc	ubuntu-15.10-standard	15.10-1	Ubuntu Wily (standard)
lxc	archlinux-base	2015-2…	ArchLinux base image.
lxc	centos-7-default	20160205	LXC default image for centos 7 (20160205)
lxc	ubuntu-12.04-standard	12.04-1	Ubuntu Precise (standard)
lxc	debian-8.0-standard	8.4-1	Debian 8.0 (standard)
lxc	ubuntu-14.04-standard	14.04-1	Ubuntu Trusty (standard)
lxc	ubuntu-16.04-standard	16.04-1	Ubuntu Xenial (standard)
lxc	centos-6-default	20160205	LXC default image for centos 6 (20160205)
lxc	gentoo-current-default	20160523	LXC default image for gentoo current (201…
Section: turnkeylinux (99 Items)			

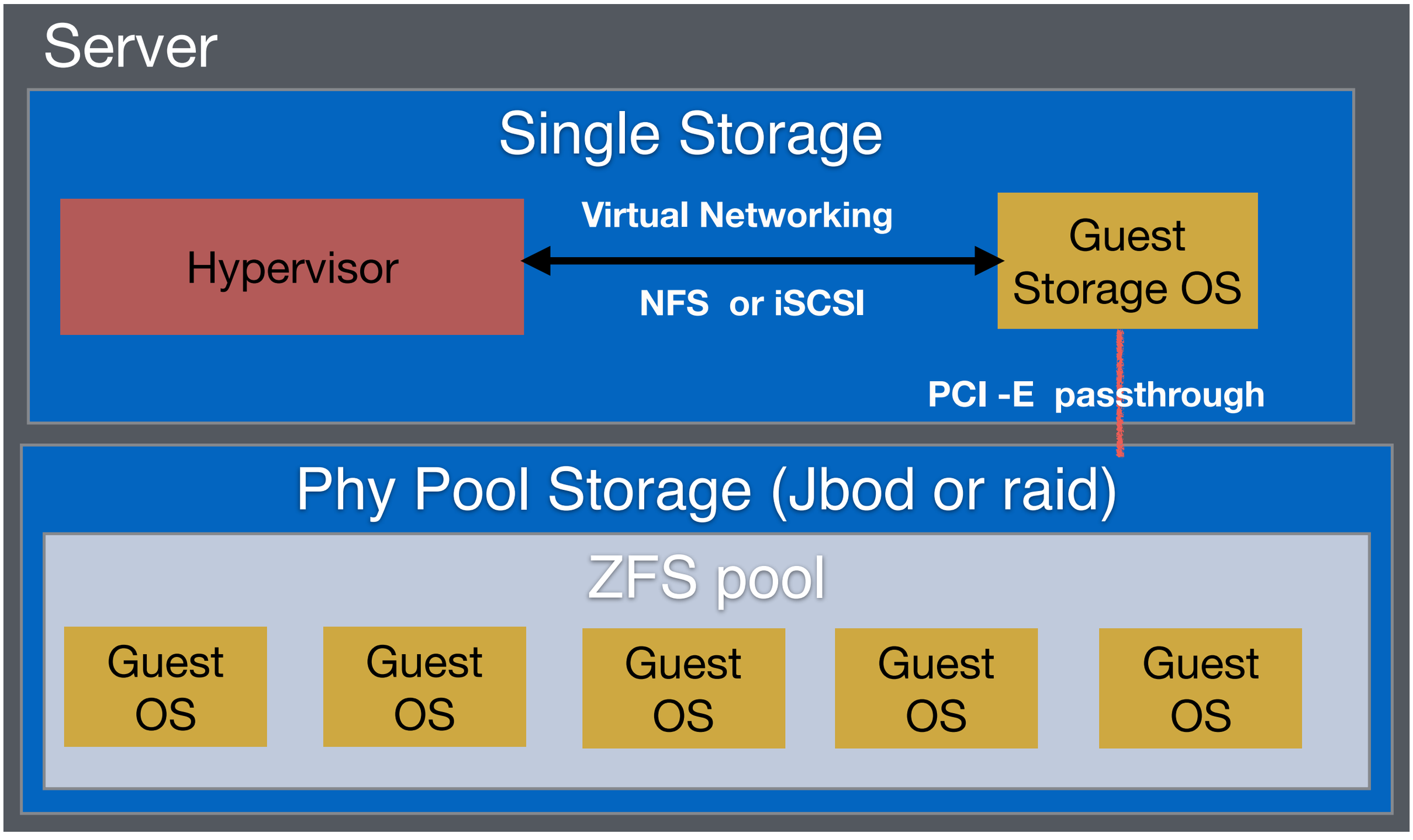
Download

Proxmox VE VM畫面

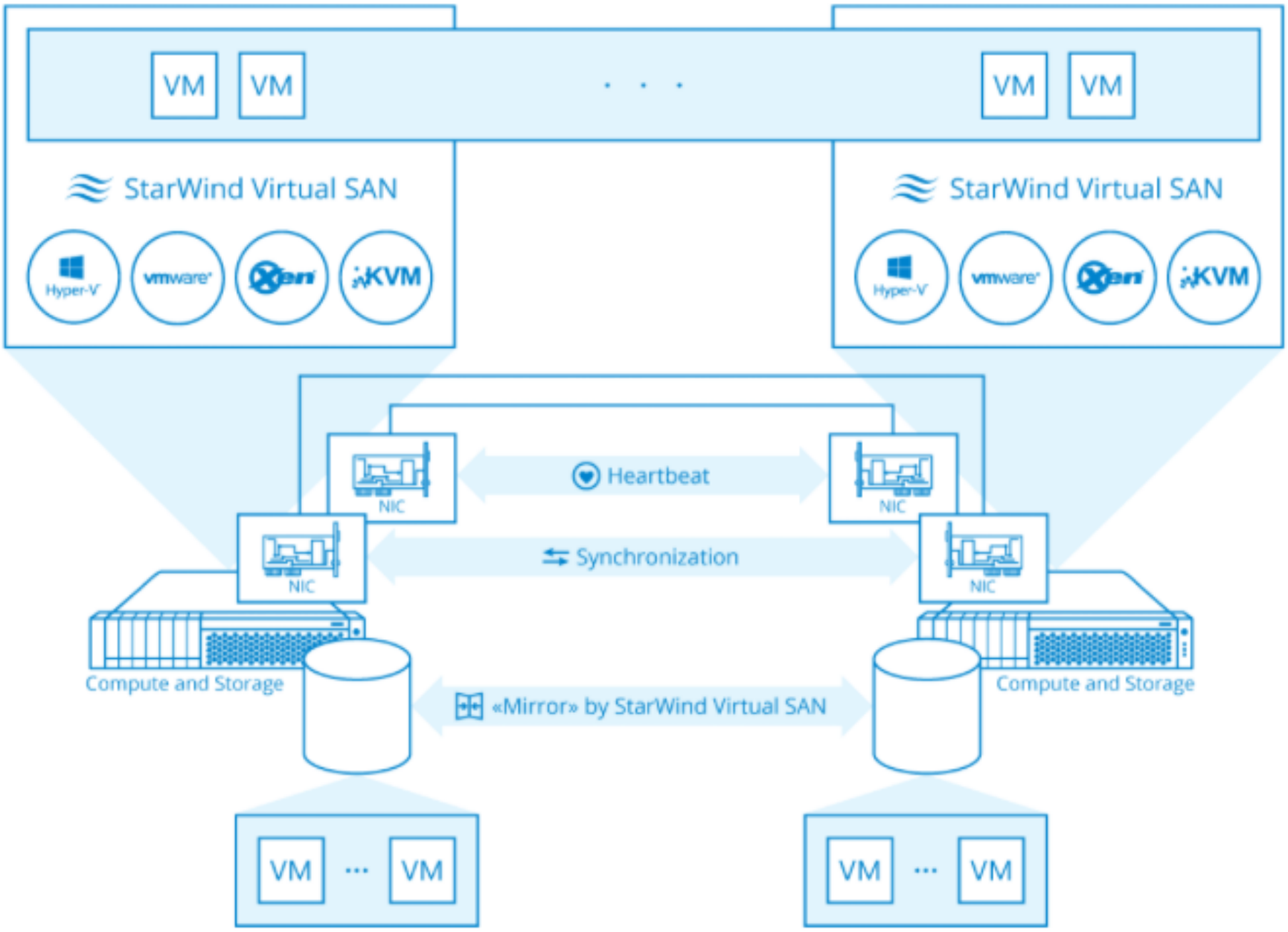
其中一台VM的使用狀態



虛擬化與Storage 整合架構



HA 怎實現的


VM
VM
VM
VM
StarWind Virtual SAN
StarWind Virtual SAN
Hyper-V
vmware
Xen
KVM
Hyper-V
vmware
Xen
KVM
Heartbeat
Synchronization
«Mirror» by StarWind Virtual SAN
NIC
NIC
NIC
NIC
Compute and Storage
Compute and Storage
VM
VM
VM
VM

實戰芭樂價硬體時代
硬體組裝與調教指南

如何挑選性價比最高的硬體主機

1. 以品牌**Server**優先，先查主機的 **Maintenance Service Guide**，確認該主機支援的 **CPU**、**DRAM**規格，然後尋找性價比最高的合用零件

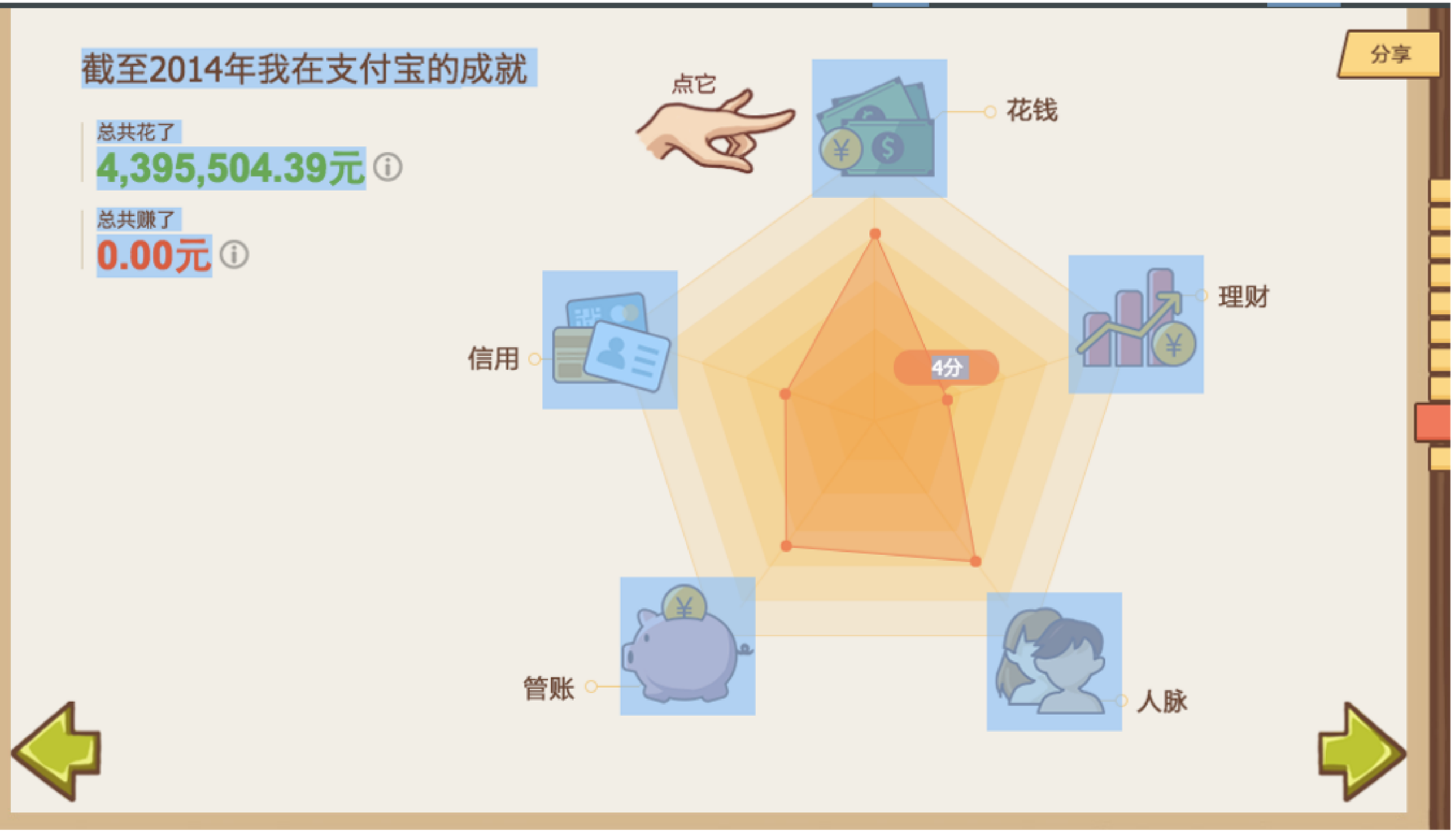
2. 自己搭配非原廠**CPU**、**DRAM**、**HDD**，但品質上得注意

3. 零件集齊之後，安裝 **Win Server 2012 R2**，把所有韌體更新至最新版以及安裝驅動，以及**MSM**陣列管理程式

4. 上虛擬化之前，**Storage**一定要測 **IOPS** 確保基底沒有問題

有聽說過淘寶嘛?

淘寶會讓你花上很多錢


截至2014年我在支付宝的成就
总共花了
4,395,504.39元
总共赚了
0.00元
分享
点它
花钱
理财
信用
4分
管账
人脉

東市買駿馬 西市買鞍韉

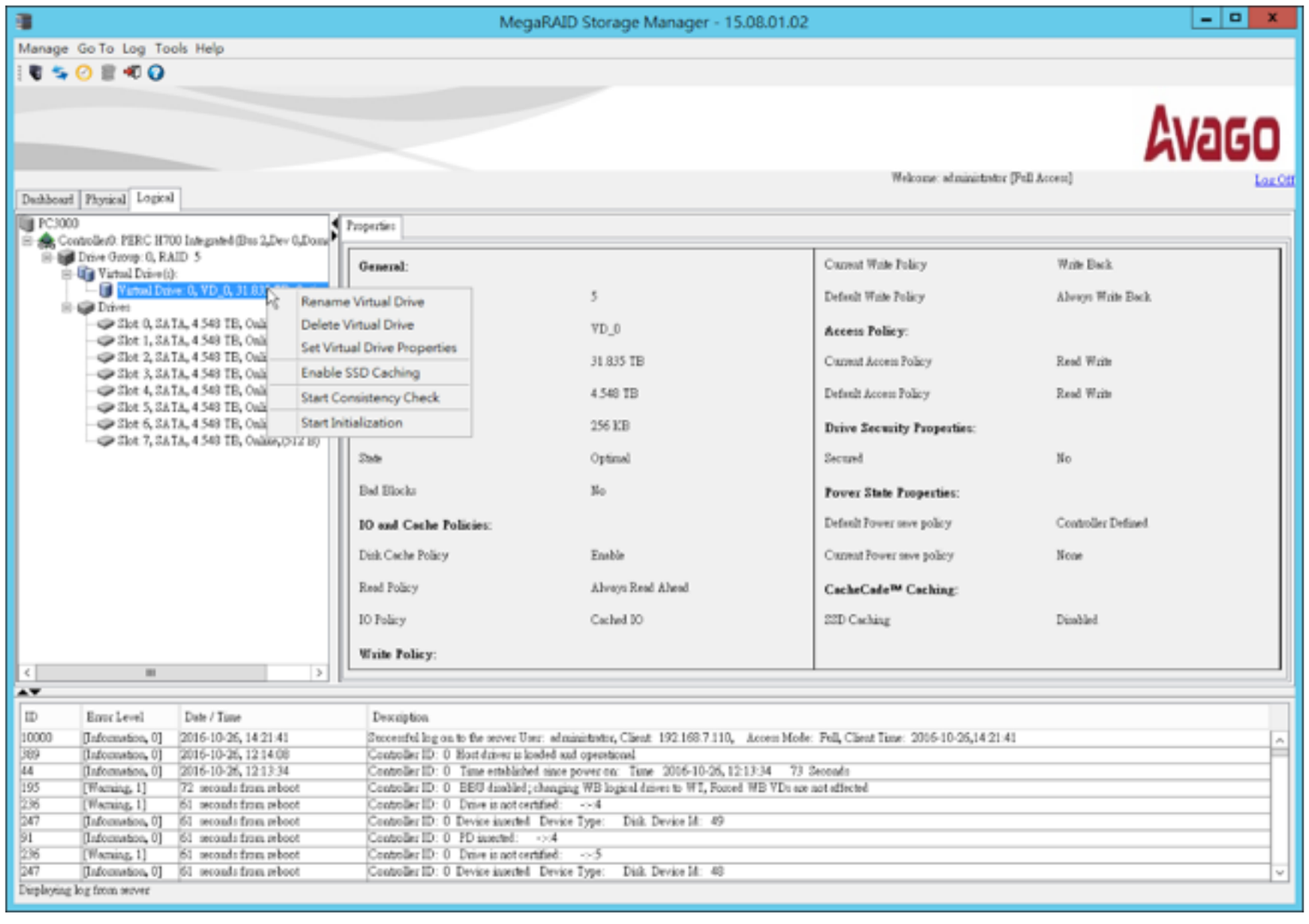
一、近新 HP ML380P G8 準系統= RMB 3000
(9個PCI-E 3.0 LOT、4 1000M、內建 ILO 4 IP KVM、DUAL 750W POWER、帶 6 BAY SLOT)
二、CPU E5-2670 2.66GHZ 8 CORE *2 正式版 RMB 500*2= RMB 1000
三、8G ECC DDR3 RMB 85 * 24，192GB = RMB 2040
四、昂貴的第二CPU 散熱片 = RMB 500
五、IBM M5014 陣列卡 = RMB 500
六、HP 3.5 硬碟框架 RMB 65 *6 = RMB 390
七、4TB SATA HDD*7 NT 3800 *7 = NTD 26600
八、APPLE AIR PCI-E SSD 128G*3 =NT 9000
九、2 PORT 10G 光纖網路卡 =NT 2500

RMB 3000+1000+2040+465+500+390+5000+運費800 =RMB 7730
總價 (7730*5) +26600 +9000 + 2500= NT 76750

整個硬體成本

HP ML380 G8 5U DUAL POWER
(9個PCI-E 3.0）
E5-2670 V1 2.66GHZ 8 CORE *2
192GB DRAM
384 GB PCI-E SSD,4TB *1　OS
IBM M5014 4TB *6　(20TB) RAID
總共NT 76750

MSMMegaRAID Storage Manager)陣列管理程式



如果用了硬體陣列卡,MSM是必備的.一定要安裝好.這樣可以設定陣列、檢查狀態、甚至調整參數，不需要進入HBA的BIOS裡進行設定，尤其有大量磁碟的時候也較為方便管理

ZFS Storage pool速度測試

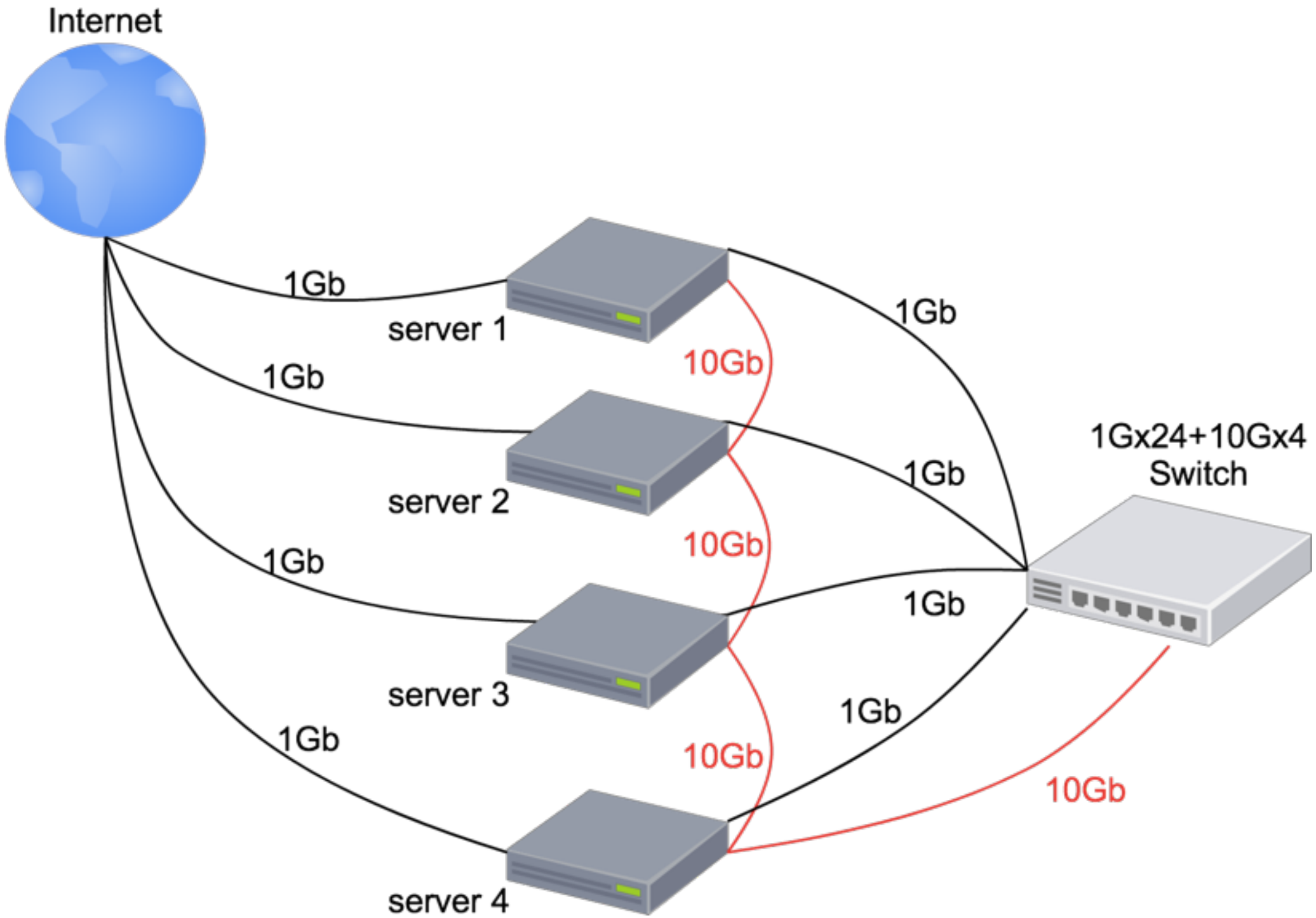
Apple PCI-e SSD 128G (ZFS) *3

```
Version: 1.03e
root@proxmox:~# bonnie++ -u root -r 1024 -s 131072 -d /AppleSSD/ -f -b -n 1
Using uid:0, gid:0.
Writing intelligently...done
Rewriting...done
Reading intelligently...done
start 'em...done...done...done...
Create files in sequential order...done.
Stat files in sequential order...done.
Delete files in sequential order...done.
Create files in random order...done.
Stat files in random order...done.
Delete files in random order...done.
Version 1.03e       ------Sequential Output------ --Sequential Input- --Random-
                    -Per Chr- --Block-- -Rewrite- -Per Chr- --Block-- --Seeks--
Machine        Size K/sec %CP K/sec %CP K/sec %CP K/sec %CP K/sec %CP  /sec %CP
proxmox        128G           886748  94 407263  57           1519413  77  3639  17
                    ------Sequential Create------ --------Random Create--------
                    -Create-- --Read--- -Delete-- -Create-- --Read--- -Delete--
               files  /sec %CP  /sec %CP  /sec %CP  /sec %CP  /sec %CP  /sec %CP
                   1   905   8 +++++ +++  1316  11   712   6 +++++ +++   544   5
proxmox,128G,,,886748,94,407263,57,,,1519413,77,3639.3,17,1,905,8,+++++,+++,1316,11,712,6,+++++,+++,544,5
root@proxmox:~#
```

RAID 5(ZFS) 4TB *6

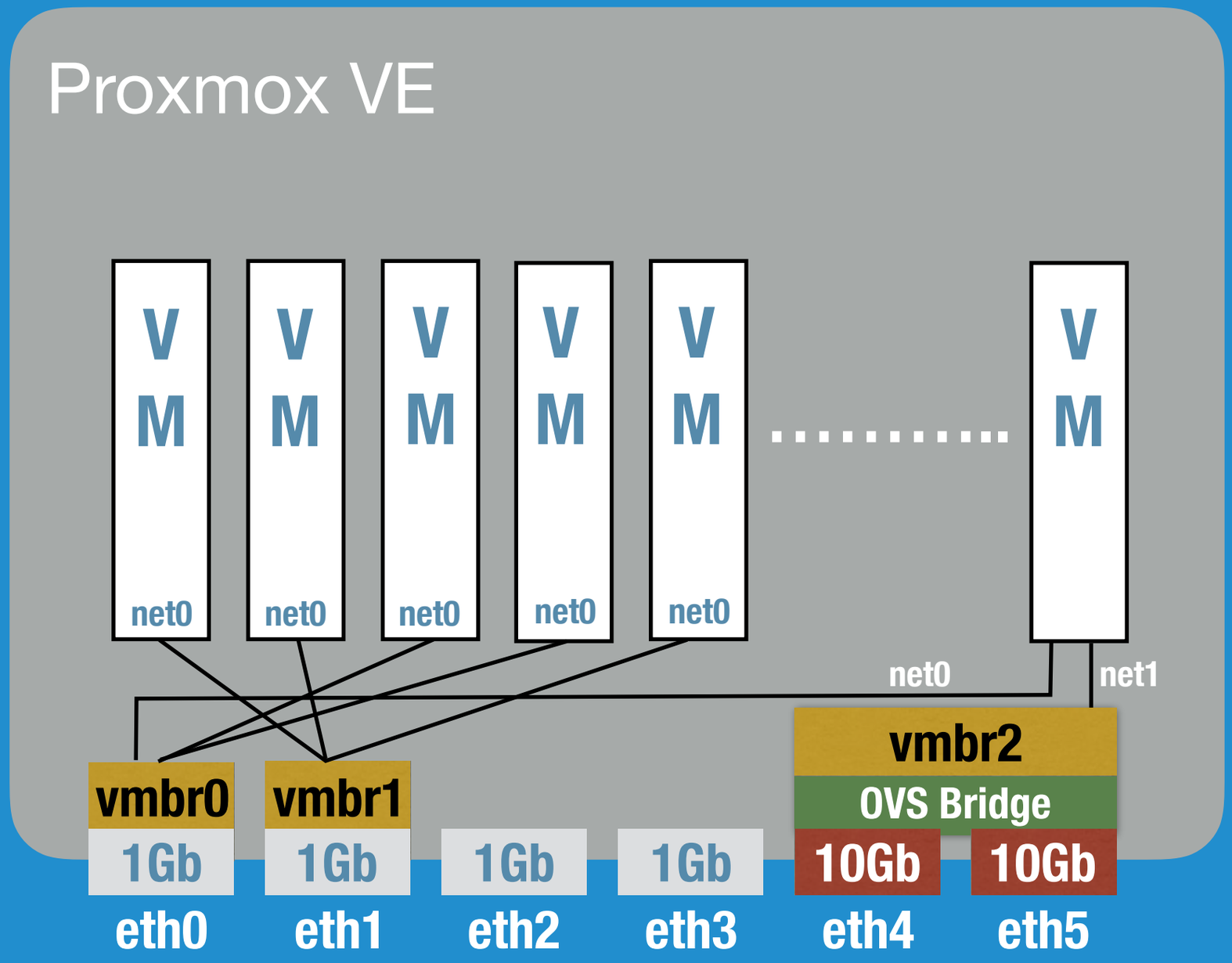
```
Delete files in random order...done.
Version 1.03e       ------Sequential Output------ --Sequential Input- --Random-
                    -Per Chr- --Block-- -Rewrite- -Per Chr- --Block-- --Seeks--
Machine        Size K/sec %CP K/sec %CP K/sec %CP K/sec %CP K/sec %CP  /sec %CP
proxmox         64G           599558  69 365373  55           2763113 100 +++++ +++
                    ------Sequential Create------ --------Random Create--------
                    -Create-- --Read--- -Delete-- -Create-- --Read--- -Delete--
               files  /sec %CP  /sec %CP  /sec %CP  /sec %CP  /sec %CP  /sec %CP
                   1 +++++ +++ +++++ +++ +++++ +++ +++++ +++ +++++ +++ +++++ +++
proxmox,64G,,,599558,69,365373,55,,,2763113,100,+++++,+++,1,+++++,+++,+++++,+++,+++++,+++,+++++,+++,+++++,+++,+++++,+++
```

窮人沒有多port 10G交換器的方法



net * 為虛擬機的虛擬網卡介面

vmbr * 為實體網卡與虛擬資源的橋接介面



規劃：

eth0 內部區網連線

eth1 對外服務網際網路連線

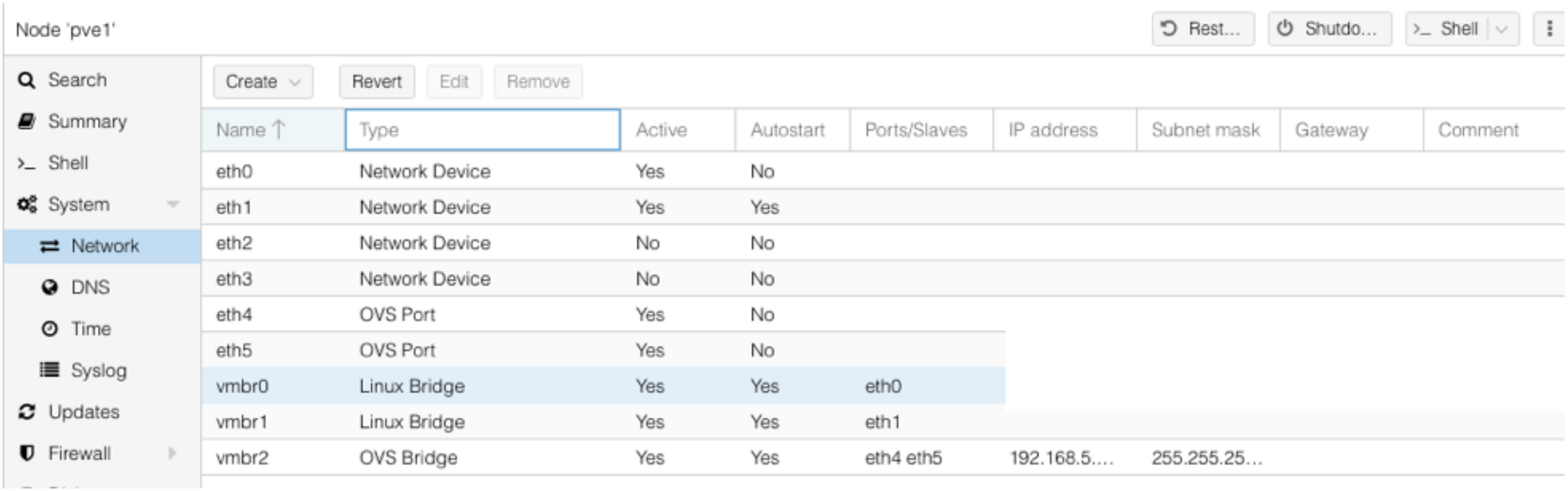
eth3 備用

eth4 備用

eth5
eth6 } 串成 **OVS Bridge** 在內部點對點傳輸，高速傳輸及備份使用

虛擬交換機 Vswitch

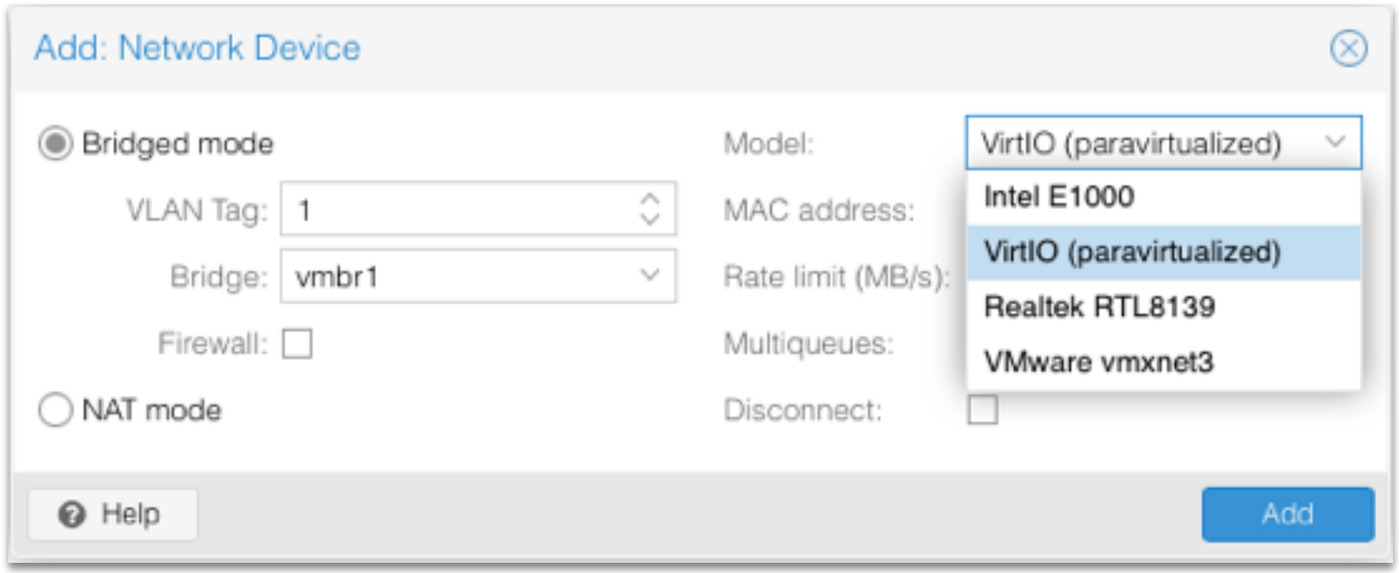
主機上的網卡，一共有內建四個 1GbE 及一張雙孔 10GbE，建立了兩個 Linux 橋接供一般VM使用，另外還使用了vSwitch 在10GbE上組成環形網路

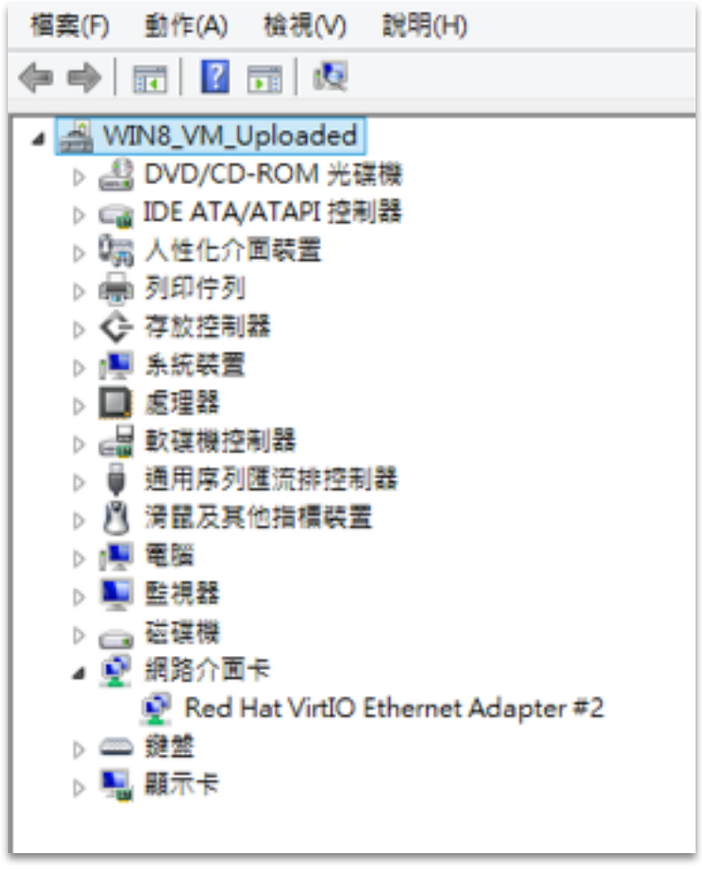


在 Proxmox VE 安裝 Open vSwitch 套件之後在 vmbr2 使用OVS Bridge 把兩個 10Gb 網孔橋接共通，這樣就可以每台伺服器透過 10G 連結，進而省下 10G switch 的費用

KVM 驅動影響效能

VM內Lan 跟Storage驅動程式必須相對應才能跑出 10Gb 的效能





VM設定網卡跟Storage的時候，Model需選擇為VirtIO，才能發揮效能

虛擬作業系統內也需要安裝驅動程式才能對應運作讓**VM與Hypervisor**傳輸效能大幅提昇。

PVE-zsync (ZFS)

開機狀態複製VM

pve-zsync 來源VM 目標儲存IP:儲存池　　名稱　　最小傳送量(Byte)

```
root@pve1:~# pve-zsync sync --source 110 --dest 192.168.7.203:RAID --verbose --name openportal --limit 102400
send from @ to RAID/vm-110-disk-1@rep_openportal_2016-10-28_14:54:41 estimated size is 1.90G
total estimated size is 1.90G
```

開始傳送過程

```
root@fmt-pve-07:~# pve-zsync create --source 100 --dest 10.0.104.201:fmt-01-nvme1/thinbackup2 --verbose
 --maxsnap 7 --name STHwp1bak
send from @ to p3700400gb/vm-100-disk-1@rep_STHwp1bak_2016-01-05_12:38:45 estimated size is 40.3G
total estimated size is 40.3G
TIME        SENT   SNAPSHOT
12:38:47    313M   p3700400gb/vm-100-disk-1@rep_STHwp1bak_2016-01-05_12:38:45
12:38:48    622M   p3700400gb/vm-100-disk-1@rep_STHwp1bak_2016-01-05_12:38:45
12:38:49    905M   p3700400gb/vm-100-disk-1@rep_STHwp1bak_2016-01-05_12:38:45
12:38:50   1.19G   p3700400gb/vm-100-disk-1@rep_STHwp1bak_2016-01-05_12:38:45
```

PVE-Sync Status

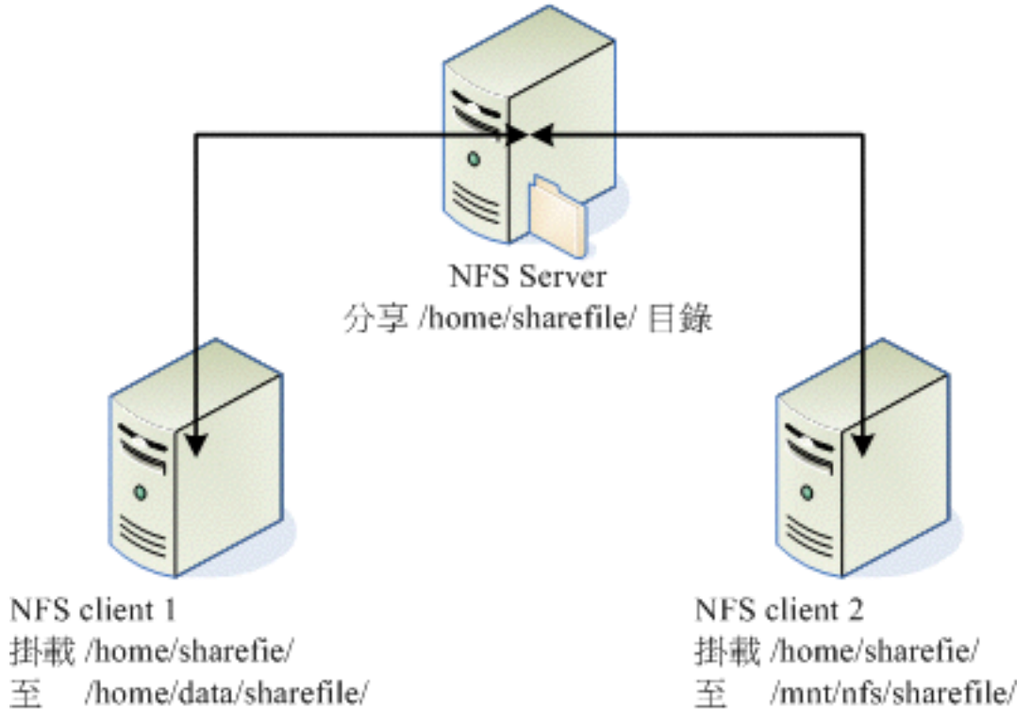
```
root@fmt-pve-07:~# pve-zsync list
SOURCE                  NAME                    STATE     LAST SYNC           TYPE  CON
100                     STHwp1bak               ok        2016-01-07_07:30:01 qemu  ssh
101                     STHf1bak                ok        2016-01-07_07:30:03 qemu  ssh
root@fmt-pve-07:~# pve-zsync status
SOURCE                  NAME                    STATUS
100                     STHwp1bak               ok
101                     STHf1bak                ok
root@fmt-pve-07:~#
```

NFS備援機制

NFS Server 架設

NFS為 Network FileSystem 的簡稱，它的目的就是想讓不同的機器、不同的作業系統可以彼此分享檔案

類似Samba Server 不過NFS在Proxmox有內建且支援性佳且能透過UI設定，因此我們在伺服器端採用NFS + Samba混合使用來處理



NFS Server 範例

Proxmox +NFS備份計畫

NFS備援機制

10GbE 網卡傳輸

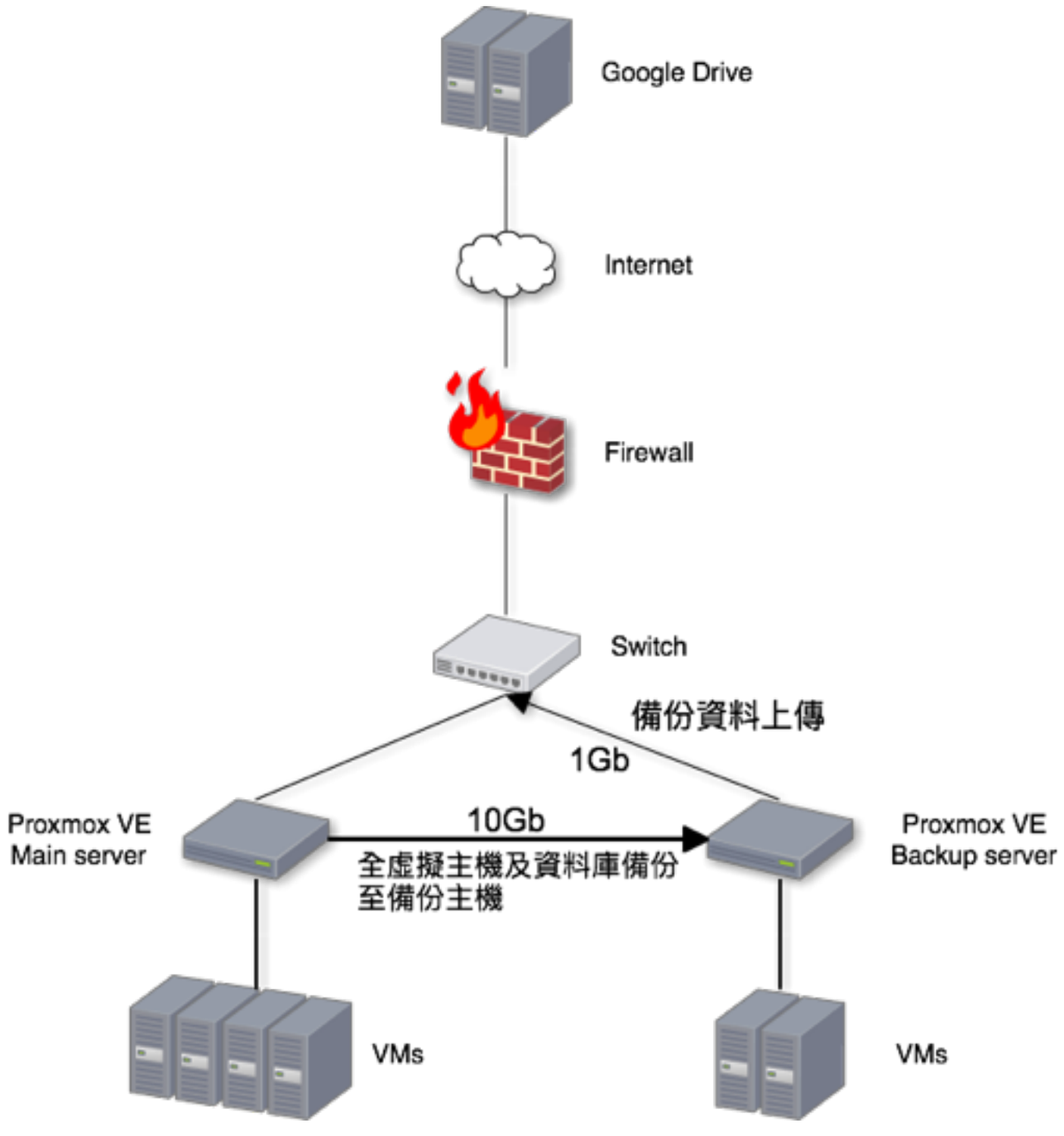
NFS Server 端設定範例

目標儲存池　讀寫權限

/RAID/NFS *(rw,sync,no_root_squash)

ID ↑	类别	内容	路径/目标	共享	生效
PVE2	NFS	VZDump 备份文件, …	/mnt/pve/PVE2	是	是
RAID	ZFS	磁盘映像, Container		否	是
local	目录	VZDump 备份文件, …	/var/lib/vz	否	是
local-lvm	LVM…	磁盘映像, Container		否	是

異地備份 - Google Drive 無限空間



透過一台VM進行vzdump ，備份資料庫、NVR以及VM儲存於本機，透過10Gb傳送到備份主機，再使用備份主機中的VM執行進行排程備份。

大家都不想看到的...但還是有可能

ACELab®

CERTIFICATE

OF PARTICIPATION

THIS IS TO CERTIFY THAT

张道弘

ZHANG DAOHONG

Has Participated In Training Seminar

«PC-3000 RAID
DATA RECOVERY TRAINING»

On the 11th of October 2016

Beijing, China

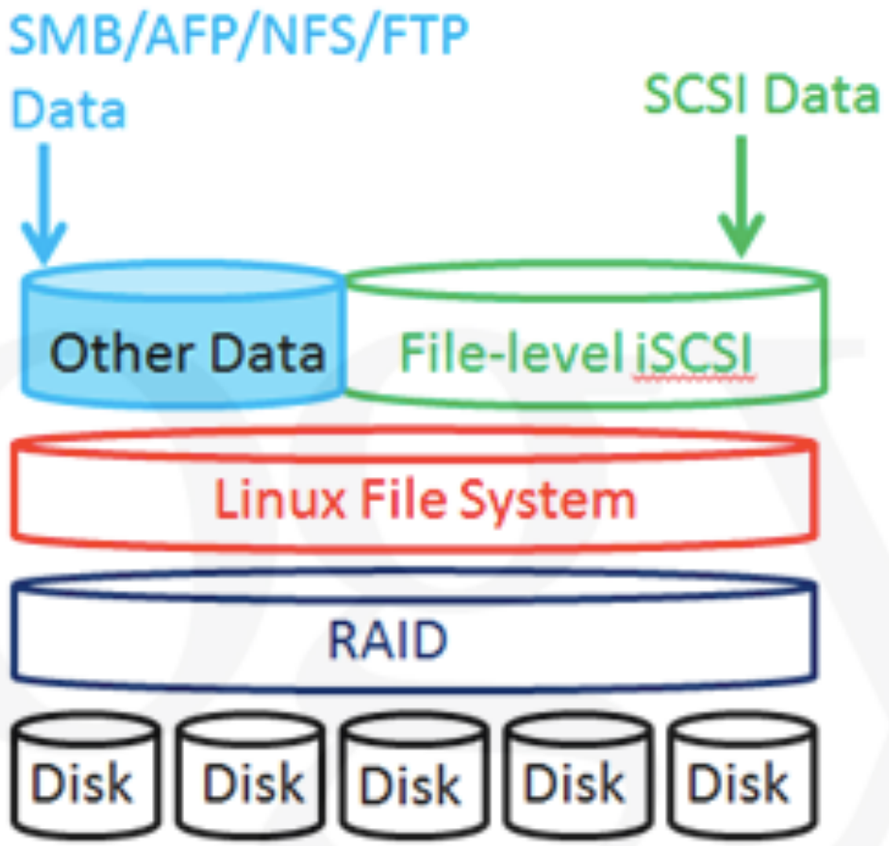
«АСЕ Лаборатори»

ELENA SHULGA
CEO
ACE LABORATORY LTD.

SDS 架構圖

當右圖四個儲存層面
若其中一層發生問題
資料則無法讀取

如果還要拯救資料時
必須從最底層處理並且一一解決每層.



HDD Level

這是一顆經由系統整合廠商轉送過來的 IBM DS3400 Cluster Storage 內的光纖介面硬碟,有做Raid 5備援,沒想到一次二顆都損壞.
單一FC 硬碟與raid 控制器接口其實不是想像中使用光通道通訊,實際上是 Fibre Channel electrical 電子串口訊號,此串口訊號其實非常接近於SATA,只是協議上有所不同.
檢查方面使用實驗室訂製光電轉換板(SAN Storage HBA 光訊號轉換成串口訊號再轉成SCA 25頭連接光纖硬碟.
才可以做
1.硬碟硬體狀況檢測. PCB ,馬達,磁頭臂組.
2.FC硬碟韌體層檢測
3.使用特製化軟體轉換光纖硬碟專有的多的ECC扇區.520 –>512 . 鏡像後得出數據.

RAID 5 算法與排列不透過硬體Raid或系統 Mount 儲存內容

D0 Xor D1 Xor D2 =P0
P0

若
可用
求得

以下以四顆

	硬碟0	硬碟1	硬碟2	硬碟3
條帶0	D0	D1	D2	P0
條帶1	D3	D4	P1	D5
條帶2	D6	P2	D7	D8
條帶3	P3	D9	D10	D11
條帶4	D12	D13	D14	P4

RAID 5 演算與排列

第一要先抓出是左循環或是右循環
假設
的分區可叫條帶
左循環就是
從
帶，校驗碼
P

請參考圖

	硬碟0	硬碟1	硬碟2	硬碟3
條帶0	D0	D1	D2	P0
條帶1	D3	D4	P1	D5
條帶2	D6	P2	D7	D8
條帶3	P3	D9	D10	D11
條帶4	D12	D13	D14	P4

RAID 5 校驗位置

同步
是資料寫入時一定會放在校驗碼
異步
則不管校驗碼換條帶時就直接照序寫入
disk

左循環同步

	硬碟0	硬碟1	硬碟2	硬碟3
條帶0	D0	D1	D2	P0
條帶1	D4	D5	P1	D3
條帶2	D8	P2	D6	D7
條帶3	P3	D9	D10	D11
條帶4	D12	D13	D14	P4

右循環異步

	硬碟0	硬碟1	硬碟2	硬碟3
條帶0	P0	D0	D1	D2
條帶1	D3	P1	D4	D5
條帶2	D6	D7	P2	D8
條帶3	D9	D10	D11	P3
條帶4	P4	D12	D13	D14

右

	硬碟0	硬碟1	硬碟2	硬碟3
條帶0	P0	D0	D1	D2
條帶1	D5	P1	D3	D4
條帶2	D7	D8	P2	D6
條帶3	D9	D10	D11	P3
條帶4	P4	D12	D13	D14

要能瞭解上面P區走法，磁碟條帶分區狀況所以分別會有：　　(以下用Win image 說法,目前來講每家說法略有點不同)

左循環同步 backward parity
左循環異步 backward dynamic
右循環同步 forward partity
右循環異步 forward dynamic

如何推算陣列參數

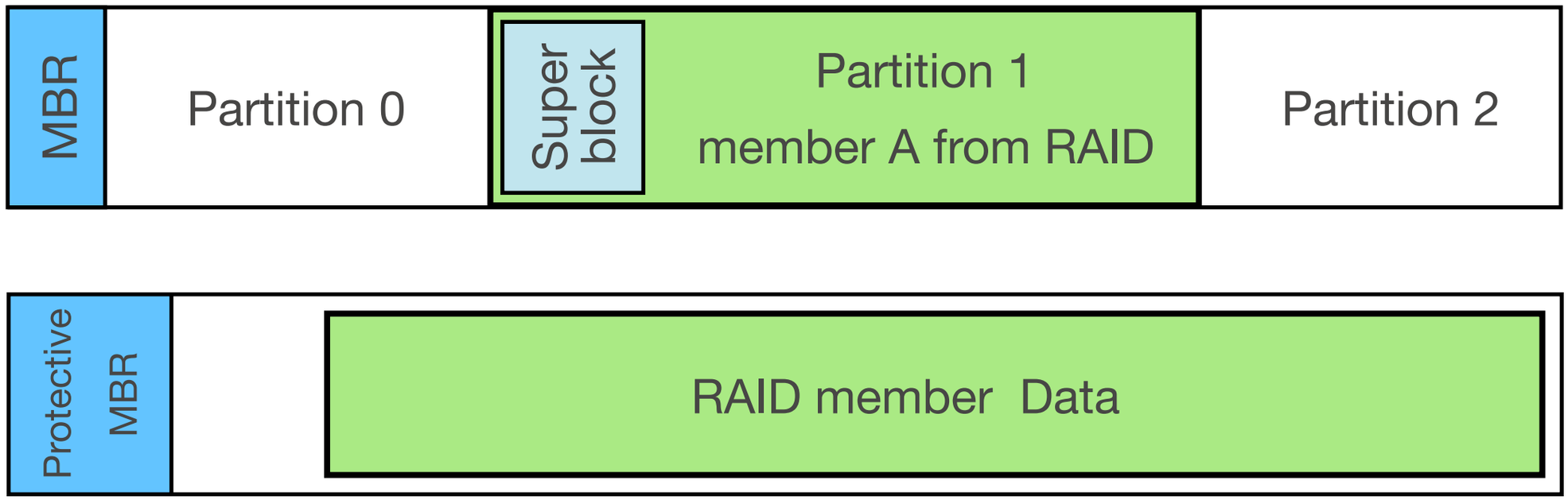
我們若能求得Raid 排列跟參數資料,就可以用軟Mount方式.比較安全性的非寫入性的拯救資料

可以從三種儲存文件結構中,推測出得Raid排列與參數.

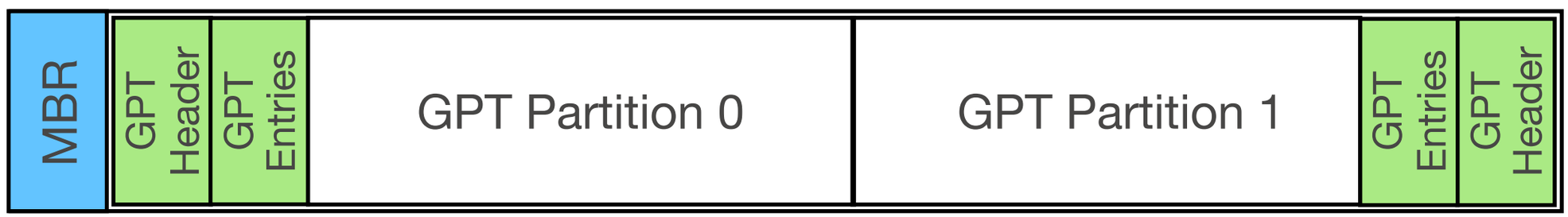
1.磁碟分區結構
2.文件系統結構
3.Raid metadata參數.

磁碟分區結構

MBR



GPT



檔案系統分析 - HFS+

LBA	Type	Size	Check	Comment
2	HFS+ Volume Header	1	0	TotBl : 20 803 544, BS : 4 096
5 894	HFS+ Volume Header	1	0	TotBl : 20 803 544, BS : 4 096
6 079	HFS+ Volume Header	1	0	TotBl : 20 803 544, BS : 4 096
6 248	HFS+ Volume Header	1	0	TotBl : 20 803 544, BS : 4 096
7 113	HFS+ Volume Header	1	0	TotBl : 20 803 544, BS : 4 096
8 206	HFS+ Volume Header	1	0	TotBl : 20 803 544, BS : 4 096
9 225	HFS+ Volume Header	1	0	TotBl : 20 803 544, BS : 4 096
10 970	HFS+ Volume Header	1	0	TotBl : 20 803 544, BS : 4 096
166 428 350	HFS+ Volume Header	1	0	TotBl : 20 803 544, BS : 4 096

整個分區是由**LBA0**開始，在每個分區的起頭，每個區域偏移是**1024 bytes**，而在結尾裡面有包含了分區大小（以**block**算）與指定區塊大小的卷標頭。這個檔案系統包含兩種標頭。然而，多重卷標頭複本還是常常可以在磁碟空間的起頭找到，這些是 **HFS+**的記錄資料，操作者必須小心避免使用到這個複本來取代了分區開頭裡真正的卷標頭

檔案系統分析 - EXT2/3/4

LBA	Type	Size	Check			Comment	
2	ExtX SuperBlock	2	0	TotBl:	1 585 642, BS:	4 096, {4B69AD91-C4FF-4B40-87F2-5B929F848BAF}, Gr#	0
262 144	ExtX SuperBlock	2	0	TotBl:	1 585 642, BS:	4 096, {4B69AD91-C4FF-4B40-87F2-5B929F848BAF}, Gr#	1
786 432	ExtX SuperBlock	2	0	TotBl:	1 585 642, BS:	4 096, {4B69AD91-C4FF-4B40-87F2-5B929F848BAF}, Gr#	3
1 310 720	ExtX SuperBlock	2	0	TotBl:	1 585 642, BS:	4 096, {4B69AD91-C4FF-4B40-87F2-5B929F848BAF}, Gr#	5
1 835 008	ExtX SuperBlock	2	0	TotBl:	1 585 642, BS:	4 096, {4B69AD91-C4FF-4B40-87F2-5B929F848BAF}, Gr#	7
2 359 296	ExtX SuperBlock	2	0	TotBl:	1 585 642, BS:	4 096, {4B69AD91-C4FF-4B40-87F2-5B929F848BAF}, Gr#	09
6 553 600	ExtX SuperBlock	2	0	TotBl:	1 585 642, BS:	4 096, {4B69AD91-C4FF-4B40-87F2-5B929F848BAF}, Gr#	25
7 077 888	ExtX SuperBlock	2	0	TotBl:	1 585 642, BS:	4 096, {4B69AD91-C4FF-4B40-87F2-5B929F848BAF}, Gr#	27

整個分區是由 **LBA0** 開始，**Ext** 檔案系統把分區分割成了一樣大小的群組，群組 **0** 在 **1024 bytes** 偏移裡包含了檔案系統的超級塊，例如：它位在分區開始的 **1024 bytes** 裡。群組 **1、3、5、7、9、25、27**也包含了超級塊，但在這裡偏移是 **0 byte**。基於這些超級塊，可以瞭解到分區大小、它的識別碼以及超級塊所位在的群組號碼。當需要區分不同分區上的超級塊，這個識別碼是很有幫助的。分區起始位置有時會包含許多損壞的超級塊複本並記錄在系統記錄中。

在某些狀況下，一個檔案系統會包含一個群組描述的可用表單並位在每個群取的起始區。在**hex**編輯器它看起來像是個會增加數值的表單，使用者需要做的只是要調整參數那麼成長的幅度就會保持住。

檔案系統分析 - BTRFS

LBA	Type	Size	Check	Comment
128	BTRFS SuperBlock	6	0	{13D55344-4BF4-ED4B-BBC0-46866BF160CA}, TotalBytes: 30 073 159 680
131 072	BTRFS SuperBlock	6	0	{13D55344-4BF4-ED4B-BBC0-46866BF160CA}, TotalBytes: 30 073 159 680

整個分區是由 **LBA0** 開始，**BtrFS** 超級塊位在以下的偏移：**64KB**、**64MB**、**256GB**、**1 PB**，不過這是在偏移沒有超過分區大小時的狀況。一個超級塊會包含分區大小。**BtrFS** 可包含數個磁碟及 **RAID** 功能。超級塊會指出 **BtrFs** 裡的磁碟數量以及現在的磁碟編號。

檔案系統分析 - VMFS

LBA	Type	Size	Check	Comment
2 048	VMFS Volume Info	1	0	{4D08C667-0342-46DB-F31E-001F29609656} ETIUSA Ultras
38 912	VMFS FS Info	0	0	{4D08C66D-F435-840F-8A14-001F29609656}, {4D08C667-C
40 960	VMFS HeartbeatRecord	0	0	{00000000-0000-0000-0000-000000000000} 3 145 728
45	VMFS HeartbeatRecord	0	0	{00000000-0000-0000-0000-000000000000} 3 146 240
40 962	VMFS HeartbeatRecord	0	0	{00000000-0000-0000-0000-000000000000} 3 146 752
40 963	VMFS HeartbeatRecord	0	0	{00000000-0000-0000-0000-000000000000} 3 147 264
40 964	VMFS HeartbeatRecord	0	0	{00000000-0000-0000-0000-000000000000} 3 147 776
40 965	VMFS HeartbeatRecord	0	0	{00000000-0000-0000-0000-000000000000} 3 148 288
40 966	VMFS HeartbeatRecord	0	0	{00000000-0000-0000-0000-000000000000} 3 148 800
40 967	VMFS HeartbeatRecord	0	0	{00000000-0000-0000-0000-000000000000} 3 149 312
40 968	VMFS HeartbeatRecord	0	0	{00000000-0000-0000-0000-000000000000} 3 149 824
40 969	VMFS HeartbeatRecord	0	0	{00000000-0000-0000-0000-000000000000} 3 150 336
40 970	VMFS HeartbeatRecord	0	0	{00000000-0000-0000-0000-000000000000} 3 150 848
40 971	VMFS HeartbeatRecord	0	0	{00000000-0000-0000-0000-000000000000} 3 151 360

整個分區是由**LBA0**開始的話，**VMFS** 可能會由多個延伸部分組成，這些延伸部分像**JBOD**一樣構成一般的空間給檔案系統的資料所使用。在偏移 **1MB** 的地方包含了容量資訊結構，可以用來知道檔案系統的大小，延伸部分的數量與現用延伸所記錄的資料配額

檔案系統分析 - UFS1/2

LBA	Type	Size	Check	Comment
128	UFS2 SuperBlock	3	3	TotFr: 2 499 761, FrS: 2 048, BS: 16 384, {4E5E730D-C1D2-3FC4-0000-000000080000}
160	UFS2 SuperBlock	3	3	TotFr: 2 499 761, FrS: 2 048, BS: 16 384, {4E5E730D-C1D2-3FC4-0000-000000080000}
192	UFS 1,2 Cylinder Group	1	0	#: 0
376 512	UFS2 SuperBlock	3	3	TotFr: 2 499 761, FrS: 2 048, BS: 16 384, {4E5E730D-C1D2-3FC4-0000-000000080000}
376 544	UFS 1,2 Cylinder Group	1	0	#: 0
752 864	UFS2 SuperBlock	3	3	TotFr: 2 499 761, FrS: 2 048, BS: 16 384, {4E5E730D-C1D2-3FC4-0000-000000080000}
752 896	UFS 1,2 Cylinder Group	1	0	#: 0
1 129 216	UFS2 SuperBlock	3	3	TotFr: 2 499 761, FrS: 2 048, BS: 16 384, {4E5E730D-C1D2-3FC4-0000-000000080000}
1 129 248	UFS 1,2 Cylinder Group	1	0	#: 0

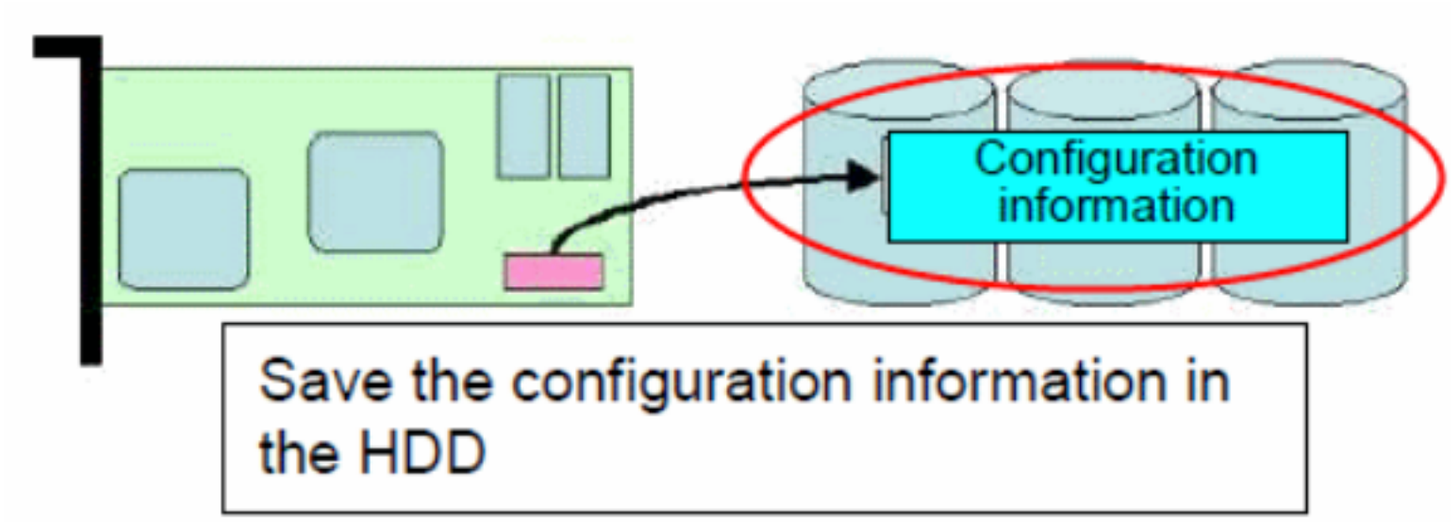
整個分區是由**LBA0**開始的話，**UFS1** 的零超級塊是位在第**16**扇區，**UFS2**的零超級塊則是位在第**128**扇區，**UFS1/2**在一個分區中會分隔成同樣大小且兩者都包含超級塊複本與**zero group**的群組，會根據包含群組編號的磁柱群組排序。**UFS1**的超級塊複本偏移是從群組到群組間變化。**UFS2**的超級塊複本則是固定在群組中的穩定偏移中

檔案系統分析 - ZFS

LBA	Type	Size	Check	Comment	Name
32	ZFS Name/Value Pair List	223	1	Pool: F69E2E6848D5E604 VDev: E98E58ED139985D7	my_zpool
264	ZFS Uberblock	0	0	23.03.2016 10:11:45	
266	ZFS Uberblock	0	0	23.03.2016 10:11:46	
268	ZFS Uberblock	0	0	23.03.2016 10:12:16	
278	ZFS Uberblock	0	0	23.03.2016 10:14:30	
280	ZFS Uberblock	0	0	23.03.2016 10:14:36	
286	ZFS Uberblock	0	0	23.03.2016 10:15:07	
288	ZFS Uberblock	0	0	23.03.2016 10:15:23	
290	ZFS Uberblock	0	0	23.03.2016 10:15:23	
293	ZFS Uberblock	0	0	23.03.2016 10:15:52	
294	ZFS Uberblock	0	0	23.03.2016 10:15:52	
296	ZFS Uberblock	0	0	23.03.2016 10:16:21	
302	ZFS Uberblock	0	0	23.03.2016 10:16:31	
308	ZFS Uberblock	0	0	23.03.2016 10:16:37	
314	ZFS Uberblock	0	0	23.03.2016 10:17:07	
318	ZFS Uberblock	0	0	23.03.2016 10:18:07	
544	ZFS Name/Value Pair List	223	1	Pool: F69E2E6848D5E604 VDev: E98E58ED139985D7	my_zpool
776	ZFS Uberblock	0	0	23.03.2016 10:11:45	
778	ZFS Uberblock	0	0	23.03.2016 10:11:46	

ZFS 使用一組 metadata 來當做”標籤”，這個組合使用大約256KB 的空間。ZFS 在分區開始包含兩組這樣的組合（在偏移 0 跟 256KB 的地方）而且還有兩組在分區結尾。每一組包含了容量池的描述（名稱/值配對表）位在偏移 16KB 的地方。從偏移 128KB 開始的地方被用在儲放最新的uberblock 版本。ZFS 可能會跨越多個磁碟並執行RAID功能，雖然它不像傳統的 RAID 那樣運作。使用者或許會遇到系統中使用傳統的 RAID 而之上有個簡單的 ZFS 功能。儲存池描述可以學習他之中有多少磁碟數量以及現有的磁碟大小。

LSI RAID METADATA 分析



對 hwHalModeStatus 變數的描述，private 和 string 的關鍵字說明了對該變數的狀態
1. 它是私有(private)的，無法被外界存取，適用在封裝應用。在VD裡面是看不到的。
2. 它是字串型態。

在 LSI RAID 控制器下對於 metadata 被包含在磁碟上記錄的兩個預留位址(reserved locations)，一個是對於該磁碟組成上的驅動器(drive)的起始 sector 上；另一個則是驅動器(drive)的結束sector上。

RAID controller上所存的設定(configuration)，允許被提出(save)和復原(restore)。。

RAID METADATA 分析 - LDM(MBR VER.)

LBA	Type	Size	Check	Comment
6	LDM: PRIVHEAD	0	0	Conf: [1 953 523 119..1 953 525 166], Data:[63..1 953 523 118]
1 953 523 121	LDM: TOCBLOCK	0	0	config [17..1 497], log [1 498..1 722]
1 953 523 136	LDM: VMDB	0	0	e2eb231d-6a05-11e6-8387-74d02b93ff8b
1 953 523 137	LDM: VBLK	1 480	1 480	Gr: 1, Dsk: 3, Vol: 3, Comp: 3, Prt: 4
1 953 524 617	LDM: KLOG	0	0	
1 953 524 618	LDM: KLOG	0	0	
1 953 524 975	LDM: PRIVHEAD	0	0	Conf: [1 953 523 119..1 953 525 166], Data:[63..1 953 523 118]
1 953 525 164	LDM: TOCBLOCK	0	0	config [17..1 497], log [1 498..1 722]
1 953 525 166	LDM: PRIVHEAD	0	0	Conf: [1 953 523 119..1 953 525 166], Data:[63..1 953 523 118]

LDM(Logical Disk Manager)，是一個Windows內部工具，最常用來佈署軟體RAID，也能用來製造分區。同樣的成員可以被包含在數個不同的陣列。

以MBR為基礎的版本使用一個獨立分區（type 0x42），會佔住幾乎整個磁碟，完全是個資料分區。Metadata儲存在這範圍之外，其中一個PRIVHEAD複本存放在磁碟空間的開頭，其餘都是存放在磁碟的結尾。

RAID METADATA 分析 - LDM(GPT VER.)

LBA	Type	Size	Check	Comment
36	LDM: TOCBLOCK	0	0	config [17..1 497], log [1 498..1 722]
51	LDM: VMDB	0	0	e2eb231d-6a05-11e6-8387-74d02b93ff8b
52	LDM: VBLK	1 480	1 480	Gr: 1, Dsk: 3, Vol: 3, Comp: 3, Prt: 4
1 532	LDM: KLOG	0	0	
1 533	LDM: KLOG	0	0	
1 890	LDM: PRIVHEAD	0	0	Conf: [34..2 081], Data:[262 178..1 953 525 133]
2 079	LDM: TOCBLOCK	0	0	config [17..1 497], log [1 498..1 722]
2 081	LDM: PRIVHEAD	0	0	Conf: [34..2 081], Data:[262 178..1 953 525 133]

GPT 使用三個分區：一個小的 metadata 分區、一個小的服務分區以及資料儲存分區 (三個中最大的)。在這個例子中，所有的 metadata 都位在他們對應的位置上。先別說 metadata 的儲存方式，其中有兩個的結構是非常有趣的：

PRIVHEAD - 它包含物理磁碟識別、群組名、RAID 所使用的分區大小及起頭、metadata 區域的大小及起頭、以及時間戳記。透過比對描繪包含磁碟大小的分區可以瞭解到 LDM 被置於 RAID 或磁碟的何處。這個資料指出LDM在何處被製造。當想要推測 metadata 可以記載組態是過時的或者嘗試修復 RAID 的時候是很有用的。

VBLK - 一類包含物理及邏輯磁碟（包含RAID）敘述的資料庫

RAID METADATA 分析 - MDADM(多重裝置，LINUX 軟體RAID)

mdadm 是 Linux 上用來製造軟體 RAID 最普遍的工具。可以使用整個磁碟或獨立分區來建立陣列。不像 LDM，在一個 RAID 中只支援使用獨立的分區或磁碟，每個 RAID 儲存它的 metadata 在每個分區或磁碟中。

這個結構保存陣列的資訊就像"Linux RAID 超級塊"。它能幫助找出組合陣列的形態、RAID 中的組成位置、組成大小以及某些組成的起頭。建立時間以及最後更新時間也會保存下來。有時這些戳記能幫助了解這系列組成的錯誤。

這結構有兩個基礎版本：0.9 與 1.x，是使用不同的儲存格式。

Ver 0.9 儲存在分區/磁碟的結尾。更特別的是，它存放在最後的完整64KB區塊。一個 RAID 的組成空間在分區/磁碟都是從扇區0開始。當RAID組成是個分區，如果沒有外部指引，就沒有辦法來精確的識別它的開頭(例如：MBR)

LBA	Type	Size	Check	Comment
1 674 784	Linux RAID Superblock 0.90	8	0	{4B1E1BBE-0000-0000-A6EB-4D9D7F99CA47} / Level 1

Ver 1.x 有三個版本，差異是在 metadata 的位置。所有 1.x 版儲存它們的偏移在分區或磁碟中，因此，可以用來尋找陣列組成的起頭及它們的大小

Ver 1.0 存在分區/磁碟的結尾

LBA	Type	Size	Check	Comment
2 050 032	Linux RAID Superblock 1	0	0	{58B948D5-B71D-D76C-2197-51454E51FF62} / Level 0

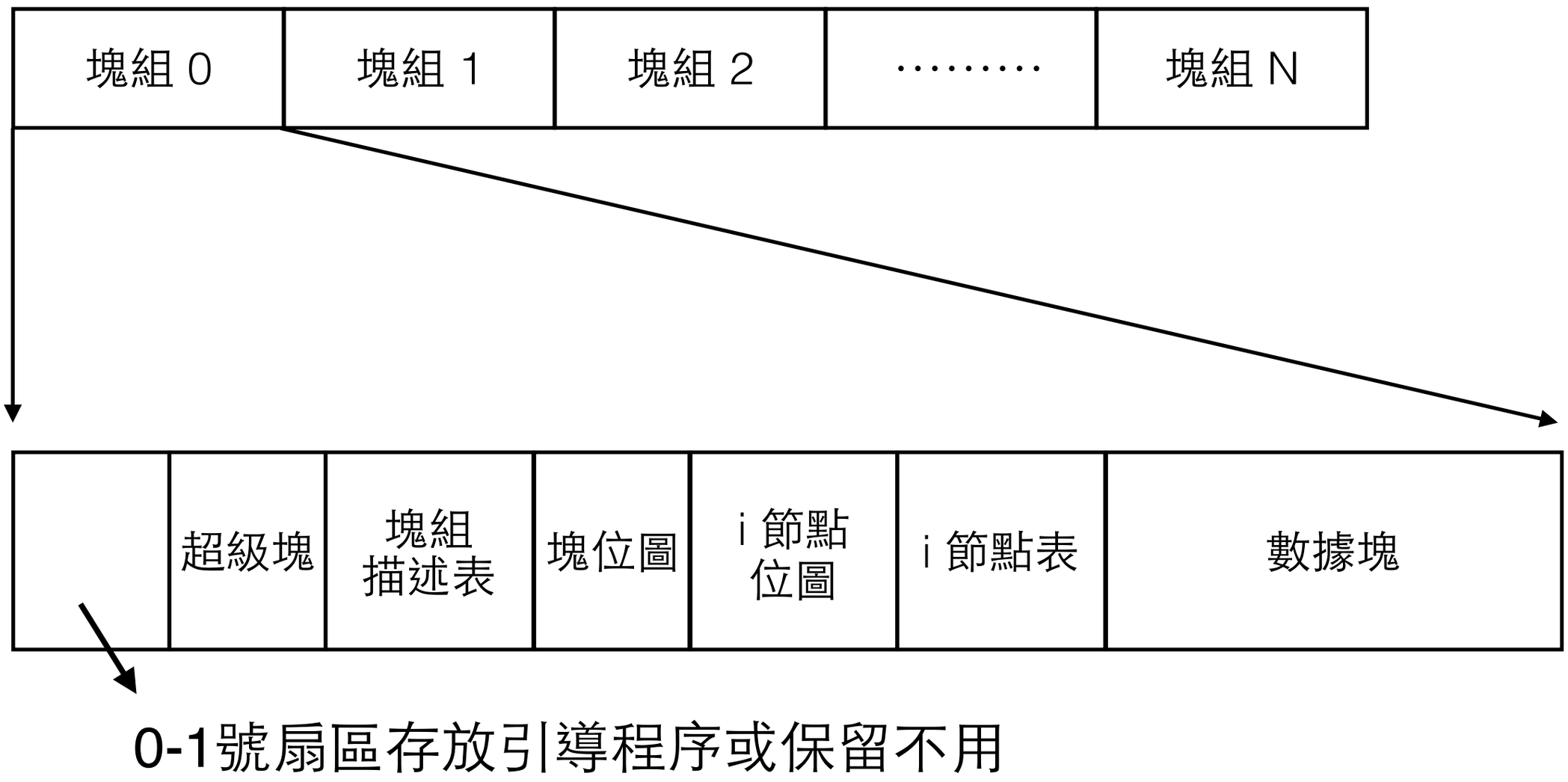
Ver 1.1 存在分區/磁碟的扇區0

LBA	Type	Size	Check	Comment
2 048	Linux RAID Superblock 1	0	0	{453B80B4-142F-3A0C-1656-1E9FB6CEF7CD} / Level 0

Ver 1.2 存在分區/磁碟的起頭扇區4KB的偏移

LBA	Type	Size	Check	Comment
2 048	Linux RAID Superblock 1	0	0	{4D236269-4C0C-17E6-4258-981E485A8B5F} / Level 0

Bitmap 形式的Filesystem結構(ext4)



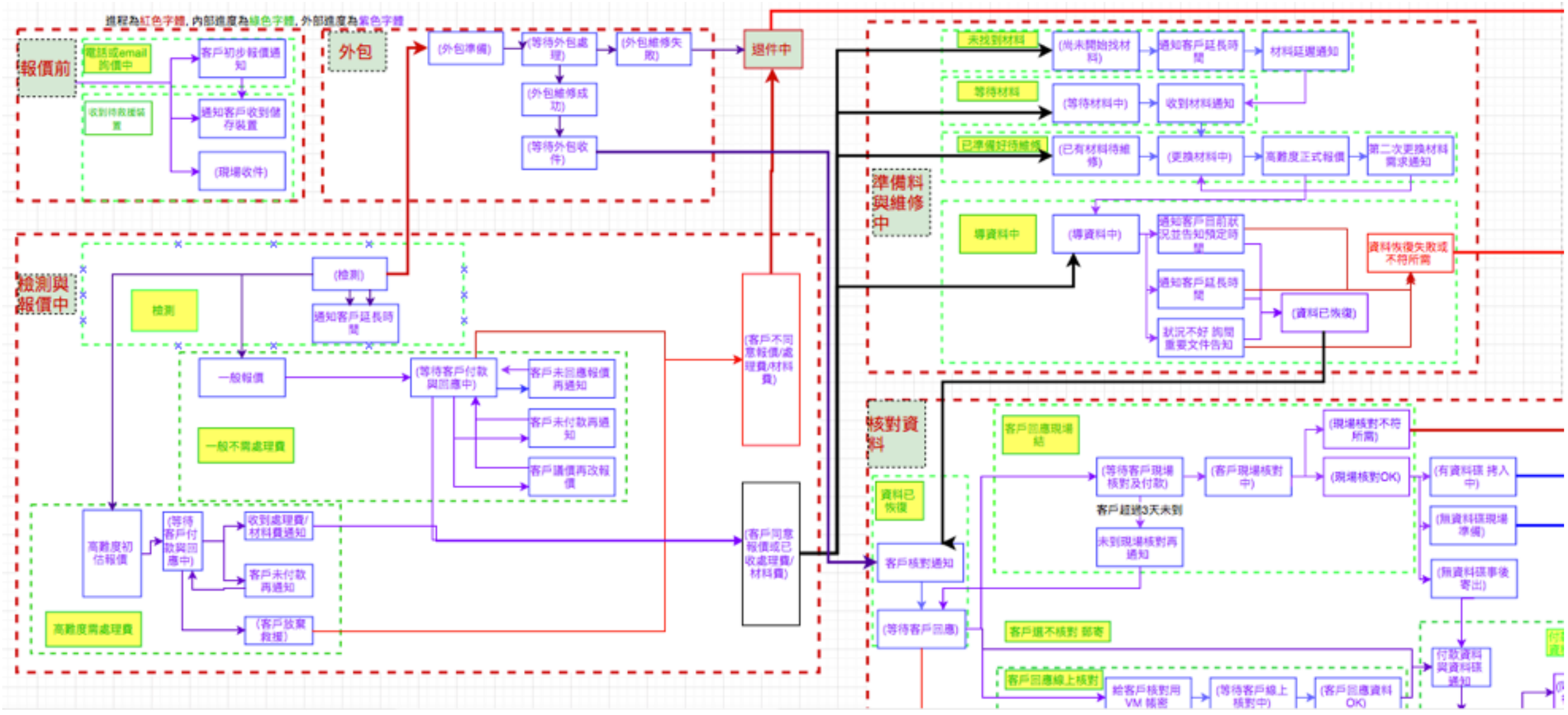
當需要分配空間時，在bitmap中查找連續的自由位，分配後，再置成已分配就可以了
釋放時，將對應位置為可分配即可

細說COW檔案系統(ZFS)

將所有需要管理的空間均勻分為不超過200個源的相同容量的小段，每個段只接受2^N。每一個分配空間叫一個metaslab。比如一個150G的全空間，除以200是750M，就向上趨近於2^N，即1GB，得到150個metaslab，每個大小為1GB。

每個metaslab用流水賬(space map)的方式記錄本metaslab的io日志，所有的釋放/分配都在這個賬本的尾部續寫。當本metaslab需要分配/釋放空間時，先按時間順序讀入這個流水賬，讀完後，就生成了本metaslab的真實分配位圖，再在內存中進行分配就可以，當達到一個事務節點時，將內存中的摘要信息(第二層)和正好在內存中順便合並後的流水賬信息回寫到磁盤上(space map entry可以視情況合並，老的記錄也可能優先合併

資料救援流程


報價前
電話或email詢價中
客戶初步報價通知
收到待救援裝置
通知客戶收到儲存裝置
(現場收件)
外包
(外包準備)
(等待外包處理)
(外包維修失敗)
(外包維修成功)
(等待外包收件)
退件中
檢測與報價中
檢測
(檢測)
通知客戶延長時間
一般報價
(等待客戶付款與回應中)
客戶未回應報價再通知
客戶未付款再通知
客戶議價再改報價
一般不需處理費
高難度初估報價
(等待客戶付款與回應中)
收到處理費/材料費通知
客戶未付款再通知
(客戶放棄救援)
高難度需處理費
(客戶不同意報價/處理費/材料費)
(客戶同意報價或已收處理費/材料費)
準備料與維修中
未找到材料
(尚未開始找材料)
通知客戶延長時間
材料延遲通知
等待材料
(等待材料中)
收到材料通知
已準備好待維修
(已有材料待維修)
(更換材料中)
高難度正式報價
第二次更換材料需求通知
導資料中
(導資料中)
通知客戶目前狀況並告知預定時間
通知客戶延長時間
狀況不好 詢問重要文件告知
(資料已恢復)
資料恢復失敗或不符所需
核對資料
資料已恢復
客戶回應現場結
(等待客戶現場核對及付款)
客戶超過3天未到
未到現場核對再通知
(客戶現場核對中)
(現場核對不符所需)
(現場核對OK)
(有資料碟 拷入中)
(無資料碟現場準備)
(無資料碟事後寄出)
客戶核對通知
(等待客戶回應)
客戶選不核對 郵寄
付款資料與資料碟寄出
客戶回應線上核對
給客戶核對用VM 賬密
(等待客戶線上核對中)
(客戶回應資料OK)

資料救援整合系統


客戶資料
送修件描述
報價/付款
詳細硬碟資料
內部材料搜尋
硬體維修記錄
軟體維修記錄
外購材料搜尋
Firmware搜尋
外包記錄
照片
客戶單號
0930423
姓名
廠牌
Hitachi
型號
HTS725050A9A364
容量
500G
界面
SATA
尺寸
2.5"
序號
數量
Preamp
Firmware
生產日期
硬碟產地
電路板號
電機號
P/N
控制IC
儲存IC
硬碟狀況
全好
ROM版本
救援目錄
Photo
優先救援
資料存放方式
備份資料存放方式
分割數量
分區格式
資料加密
儲存裝置
FW 800 RAID0
外接盒型號
Firmware路徑
導出資料路徑
導出Image路徑
打開路徑
生成文字檔
打開路徑
VHD檔應用
磁頭狀況
0
1
2
3
4
5
6
7
8
9
10
11
讀取
寫入
導出到詳細硬碟資料
故障狀況
客戶簡述
裝itunes後，無法開機
客戶備註
報告備註
filevault加密 可完整救援
檢測結果
檔案名稱結構損壞-分區表損毀，OS 無法正常掛載分區跟檔案結構
多重
維修方法
多重

歡迎一起研究精進，不吝指教

Q & A