

當按下 Proxmox PVE VM Snapshot 連動 Dell PowerStore Storage Volume 快照

幾種常見共用儲存的支援情況：

1. 支援連動（Hardware/Storage-level Snapshots）

如果您希望在 PVE 按下「快照」時，儲存設備端也同步執行快照（而非單純在虛擬磁碟檔案內做標記），目前主要有以下幾種方式：

- **ZFS over iSCSI:** 這是 PVE 最老牌的連動方式。當您在 PVE 介面建立 VM 快照時，PVE 會透過 SSH 發送指令給後端的 ZFS 儲存設備（如 TrueNAS），直接觸發 **ZFS Snapshot**。
- **協力廠商 Plugin (如 Blockbridge):** 某些企業級儲存廠商提供專用的 PVE 驅動，能讓 PVE 的快照指令直接「卸載」到 SAN 儲存設備硬體上執行，達成真正的硬體連動。
- **LVM Shared Storage (PVE 9.x 新特性預期):** 在最新的 PVE 版本中，針對傳統 SAN (iSCSI/FC) 上的 **Thick-provisioned LVM**，開發團隊正致力於改善快照支援。過去 Shared LVM 是完全不支援快照的，但目前的演進趨勢是讓 LVM 快照能在叢集環境下更穩定地運作。
- 針對 PowerStore 這種高端設備，建議採用以下架構來平衡管理與效能：

協議	PVE 快照行為	PowerStore 角色	建議配置
NFS	支援（qcow2 內部）	底層資料塊去重與壓縮	適合一般應用，管理最方便。
FC / iSCSI	需配置 LVM-thin 始支援	提供高速 IOPS 與多路徑備援	適合資料庫，快照建議由 <b>PBS</b> 處理。

/etc/systemd/system/pve-ps-monitor.service

## 注意事項與風險

- **安全性：** 腳本中含有密碼，請務必限制該檔案的讀取權限（`chmod 700`），或使用 **API Key/Token**。
- **快照清理：** 這種方式建立的 **PowerStore** 快照不會被 **PVE** 自動刪除。您必須在 **PowerStore** 上設定 **Snapshot Expiration Policy (過期策略)**，否則儲存空間會被佔滿。

**PVE 9.x** 針對 **Shared LVM (iSCSI/FC)** 的新特性，這項功能稱為 "**Snapshots as Volume-Chain**" (磁碟鏈快照)。

這項技術打破了過去「**Shared LVM** 無法快照」的限制，原理是在 **LVM** 塊設備上直接運行 **qcow2** 格式，並透過「鏈接 (Chaining)」多個 **Logical Volumes (LV)** 來達成快照。

以下是在 **PVE 9.x** 中針對 **Dell PowerStore (FC/iSCSI)** 設定新磁碟並啟用此特性的步驟：

---

### 1. 建立 **Shared LVM** 儲存空間

首先，您需要先在 **PVE** 中掛載 **PowerStore** 的 **LUN**，並將其格式化為 **Shared LVM**。

1. 在 **PowerStore** 端： 建立一個 **Volume** 並對應 (Mapping) 給 **PVE** 叢集的所有節點。
2. 在 **PVE** 網頁介面：
  - 前往 **Datacenter > Storage > Add > LVM**。
  - **ID:** 自訂名稱 (例如 **PowerStore\_LVM**)。
  - **Base storage:** 選擇您的 **iSCSI** 或 **FC** 目標。
  - **Volume Group:** 選擇 **PowerStore** 的 **LUN**。
  - **Shared:** 務必勾選 (這是叢集共用的關鍵)。
  - **Advanced (進階設定):** 勾選 "**Enable Snapshots as Volume**"

## Chains (Technology Preview)"。

注意：這是 PVE 9.x 的新功能，目前標記為技術預覽版。

---

### 2. 替 VM 新增一個支援快照的磁碟

在 Shared LVM 設定好後，新增磁碟時需要注意 **磁碟格式**：

1. 選擇您的 **VM > Hardware > Add > Hard Disk**。
2. **Storage**: 選擇剛剛建立的 PowerStore\_LVM。
3. **Format**: 這是關鍵！請選擇 **QCOW2 (QEMU image format)**。
  - 雖然底層是 LVM 塊設備，但 PVE 9.x 允許在 LVM 上直接封裝 qcow2 標頭，藉此獲得快照能力。
4. **Size**: 設定您需要的容量。

### 3. 快照是如何運作的？(與 PowerStore 的關係)

當您在這種模式下對 VM 執行快照時：

- **PVE 動作**：PVE 會在該 LVM Volume Group 中建立一個新的 LV (Logical Volume)，作為 Snapshot Delta。
- **空間佔用**：由於是 Thick-provisioned LVM，PVE 會預先分配與原始磁碟相同大小的空間。例如 100GB 的 VM 做一次快照，會瞬間佔用 200GB 的 LVM 空間。
- **PowerStore 連動**：
  - \* **自動連動**：無 (PVE 僅在 LVM 層級操作，不會告訴 PowerStore 建立硬體快照)。
  - **優點**：PowerStore 底層通常有硬體壓縮與去重複 (Deduplication)，雖然 PVE 顯示佔用了兩倍空間，但在 PowerStore 實體硬碟上，重複的資料塊並不會佔用雙倍空間。

2026-01-30 第一階段\*\*建立服務使 PVE Guest VM 按下快照時，連動 PowerStore 那 Volume 也建立快照

這是在 PVE 與 PowerStore 連動中最核心的挑戰。由於 Proxmox 的 VMID 是邏輯編號，而 PowerStore 的 Volume ID 是硬體 UUID，兩者之間沒有預設

的對應關係。

### 方案一：利用 Dell PowerStore 的 "Volume Name" (最推薦)

這是最直覺且低開發成本的方法。在 PowerStore 建立 Volume 時，將 **Volume Name** 命名為包含 VMID 的格式（例如：PVE\_DISK\_VM101\_01）。

腳本邏輯：

1. 透過 API 查詢 PowerStore，搜尋名稱包含 VM\$VMID 的所有 Volume。
2. 取得回傳的 id 欄位。

### API 查詢範例 (Python/Bash)：

Bash

# 透過名稱過濾取得 ID

```
VOLUME_ID=$(curl -k -u "$USER:$PASS"
```

```
"https://$PS_IP/api/rest/volume?name=like.*VM$VMID*" | jq -r '[0].id')
```

- **優點：** 易於維護，從 PowerStore 介面一眼就能看出磁碟屬於誰。
- **缺點：** 建立 Volume 時需手動（或自動化）遵循命名規範。

---

### 方案二：透過 WWN (World Wide Name) 硬核比對

這是技術上最精準的方法，不需要依賴命名規則。PVE 知道磁碟的 WWN，而 PowerStore 也知道。

自動化路徑：

1. 在 **PVE** 側：讀取 VM 設定檔 `/etc/pve/qemu-server/$VMID.conf`。
2. 找出設備名稱：從設定檔中抓取對應的儲存路徑，並對應到 `/dev/disk/by-id/wwn-0x...`。
3. **API 比對：** 拿著這個 WWN (去掉 0x 前綴) 向 PowerStore 查詢 wwn 屬性。

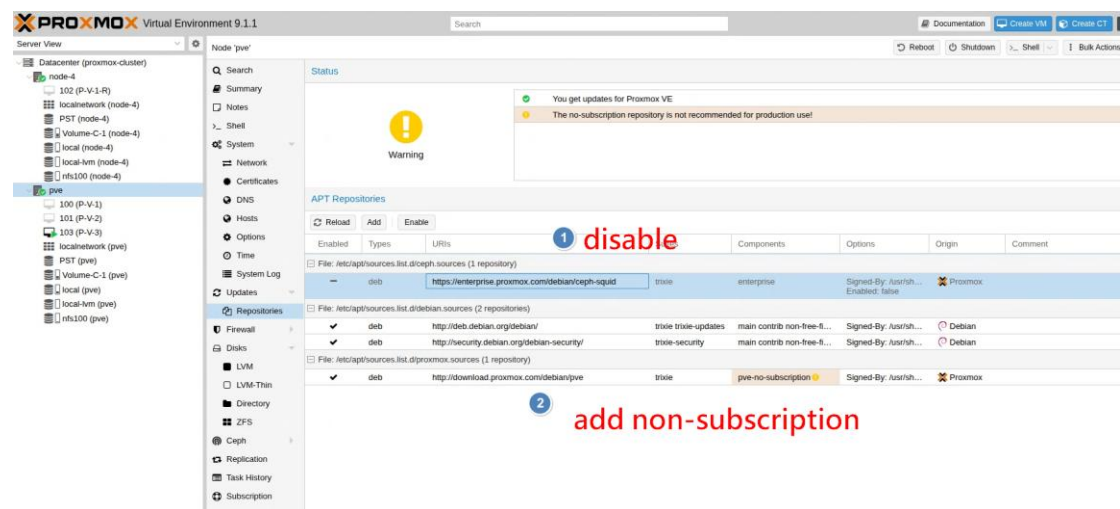
### 方案三：使用 PVE 的 "Notes" 或自定義標籤

在 PVE 的 VM 備註（Notes）中手動填入 PowerStore 的 Volume ID。

腳本邏輯：

1. 使用 `pvesh get /nodes/{node}/qemu/{vmid}/config` 取得備註欄位。
  2. 提取出 `PowerStore_ID: res_xxx` 這樣的字串。
- **優點：** 不必寫複雜的搜尋邏輯。
  - **缺點：** 人為錄入錯誤風險高，且磁碟搬移時需更新備註。

#先開要能找外部源



安裝 **jq**：腳本中解析 JSON 需要用到 **jq** 工具。在 PVE 執行：`apt update && apt install jq -y`。

**實作：PVE 與 PowerStore 連動腳本**

請將此內容儲存為 `/usr/local/bin/powerstore-snap.sh`

```
#!/bin/bash
```

```
# --- 配置區 ---
```

```
PS_IP="192.168.236.80"
```

PS\_USER="admin"

PS\_PASS="Password123!"

VMID=\$2

PHASE=\$1

TARGET\_VG="PSG-VG"

if [ "\$PHASE" == "post-snapshot" ]; then

    echo "--- [PowerStore Linkage] Starting for VM \$VMID ---"

    # 1. 取得 PSG-VG 所在的實體設備名稱 (例如 /dev/sdi)

    REAL\_PV=\$(pvs --noheadings -o pv\_name,vg\_name | grep "\$TARGET\_VG" |  
    awk '{print \$1}' | head -n 1 | tr -d '[:space:]')

    if [ -z "\$REAL\_PV" ]; then

        echo "Error: Could not find Physical Volume for \$TARGET\_VG"

        exit 1

    fi

    # 2. 取得該設備的 WWN 並進行字串清洗

    REAL\_DM=\$(readlink -f "\$REAL\_PV")

    WWN\_NAME=\$(ls -l /dev/disk/by-id/ | grep "wwn-0x" | grep "\$(basename  
\$REAL\_DM)\$" | awk '{print \$9}' | head -n 1 | tr -d '[:space:]')

    if [ -z "\$WWN\_NAME" ]; then

        echo "Error: Could not find WWN in /dev/disk/by-id for \$REAL\_DM"

```
exit 1
```

```
fi
```

```
# 去掉 wwn-0x，強制小寫，並移除所有換行或空白
```

```
CLEAN_WWN=$(echo "$WWN_NAME" | sed 's/wwn-0x//' | tr '[:upper:]' '[:lower:]' | tr -d '[:space:]')
```

```
PS_WWN="naa.${CLEAN_WWN}"
```

```
echo "Target PowerStore WWN: [$PS_WWN]" # 使用中括號檢查有無隱藏空白
```

```
# 3. 透過 API 取得 PowerStore Volume ID
```

```
# 這裡加入 -s (silent) 並確保變數被正確引用
```

```
API_URL="https://$PS_IP/api/rest/volume?select=id&wwn=eq.$PS_WWN"
```

```
PS_VOLUME_ID=$(curl -k -s -u "$PS_USER:$PS_PASS" "$API_URL" | jq -r '.[0].id' | tr -d '[:space:]')
```

```
if [ "$PS_VOLUME_ID" != "null" ] && [ -n "$PS_VOLUME_ID" ]; then
```

```
# 4. 執行 PowerStore 硬體快照
```

```
SNAP_NAME="PVE_VG_SNAP_${VMID}_${date +%Y%m%d_%H%M}"
```

```
EXPIRY=$(date -u -d "+24 hours" +"%Y-%m-%dT%H:%M:%SZ")
```

```
echo "Creating Snapshot: $SNAP_NAME (Expires: $EXPIRY)"
```

```
RESULT=$(curl -k -s -u "$PS_USER:$PS_PASS" \
```

```

-X POST
"https://$PS_IP/api/rest/volume/$PS_VOLUME_ID/snapshot" \

-H "Content-Type: application/json" \

-d "{\"name\": \"$SNAP_NAME\", \"description\": \"PVE VM $VMID
Snapshot Linkage\", \"expiration_timestamp\": \"$EXPIRY\"}")

# 簡單檢查回傳結果是否包含 id

if echo "$RESULT" | grep -q "id"; then

    echo "SUCCESS: PowerStore Hardware Snapshot created."

else

    echo "FAILED: API response: $RESULT"

fi

else

    echo "ERROR: PowerStore API could not find Volume ID for WWN
$PS_WWN"

    echo "Debug: API URL was $API_URL"

fi

fi

```

設定執行權限 `chmod +x filename`

##這還沒加入

進階叮嚀：快照的「生老病死」

這套腳本解決了「生」（建立），但您需要考慮「死」（清理）：

- **空間預警：** PowerStore 的快照雖然有點對點去重，但如果 VM 變動量大，累積過多快照會消耗儲存池空間。



- **自動清理：** 建議在 PowerStore 的管理介面上，為這些透過 API 建立的快照設定一個 **"Snapshot Policy"** 或 **"Expiration" (到期時間)**。API 指令中可以加入 `expiration_timestamp` 參數來設定自動刪除時間。

##

在 PVE 網頁介面按下「快照 (Snapshot)」按鈕時自動觸發

方案 C：使用定時監控 (Event Monitor)

這是一個比較「聰明」的做法，不需要修改任何 PVE 核心。我們可以寫一個微型背景服務 (Daemon)，監控 PVE 的快照日誌：

建立 `/usr/local/bin/pve-monitor-snap.sh`

```
#!/bin/bash
```

```
# 定義連動腳本位置
```

```
HOOK_SCRIPT="/usr/local/bin/powerstore-hook.sh"
```

```
echo "Starting PVE Snapshot Monitor (Exact Log Match) for PowerStore..."
```

```
# 監控 journal 輸出
```

```
journalctl -u pvedaemon -f -n 0 | while read line; do
```

```
    # 匹配結尾包含 qmsnapshot 且狀態為 OK 的行
```

```
    if echo "$line" | grep -q "end task" && echo "$line" | grep -q "qmsnapshot"
    && echo "$line" | grep -q "OK"; then
```

```
        # 根據您的日誌格式: <UPID>:<TASK>:<VMID>:<USER>: OK
```

```
        # 我們用冒號切分，VMID 通常在倒數第三個欄位
```

```

VMID=$(echo "$line" | awk -F: '{print $(NF-2)}')

# 安全檢查：確保 VMID 是純數字

if [[ "$VMID" =~ ^[0-9]+$ ]]; then

    echo "[$(date)] Match Found! VMID: $VMID. Triggering PowerStore
Linkage..."

    # 異步執行 Hook，避免阻塞監控迴圈

    $HOOK_SCRIPT post-snapshot "$VMID" >> /var/log/pve-
powerstore-linkage.log 2>&1 &

else

    echo "[$(date)] Warning: Could not parse VMID from line: $line"

fi

fi

done

```

設定執行權限 `chmod +x filename`

## 方案 D：直接整合至 PBS (Proxmox Backup Server)

如果您有安裝 PBS，這其實是 Dell PowerStore 最完美的搭檔。

- 在 **PowerStore** 側：設定一個 **Snapshot Rule**（例如每 15 分鐘一次），直接在儲存層級做連續保護（Continuous Data Protection）。
- 在 **PVE** 側：每天執行一次 PBS 備份到異地。

### 3. 將其建立為系統服務 (Systemd Service)

為了讓這個監控器在開機時自動啟動，且在背景持續運行，我們把它包成一個服務。

請建立 `/etc/systemd/system/pve-ps-monitor.service`：

[Unit]

Description=Proxmox Snapshot Monitor for Dell PowerStore

After=pve-cluster.service

[Service]

ExecStart=/usr/local/bin/pve-monitor-snap.sh

Restart=always

RestartSec=5

StandardOutput=append:/var/log/pve-powerstore-linkage.log

StandardError=append:/var/log/pve-powerstore-linkage.log

[Install]

WantedBy=multi-user.target

啟動服務

`systemctl daemon-reload`

`systemctl enable --now pve-ps-monitor.service`

1. 自動化追蹤：PVE 透過 WWN 自動在 PowerStore 數百個 LUN 中找到正確的對象。

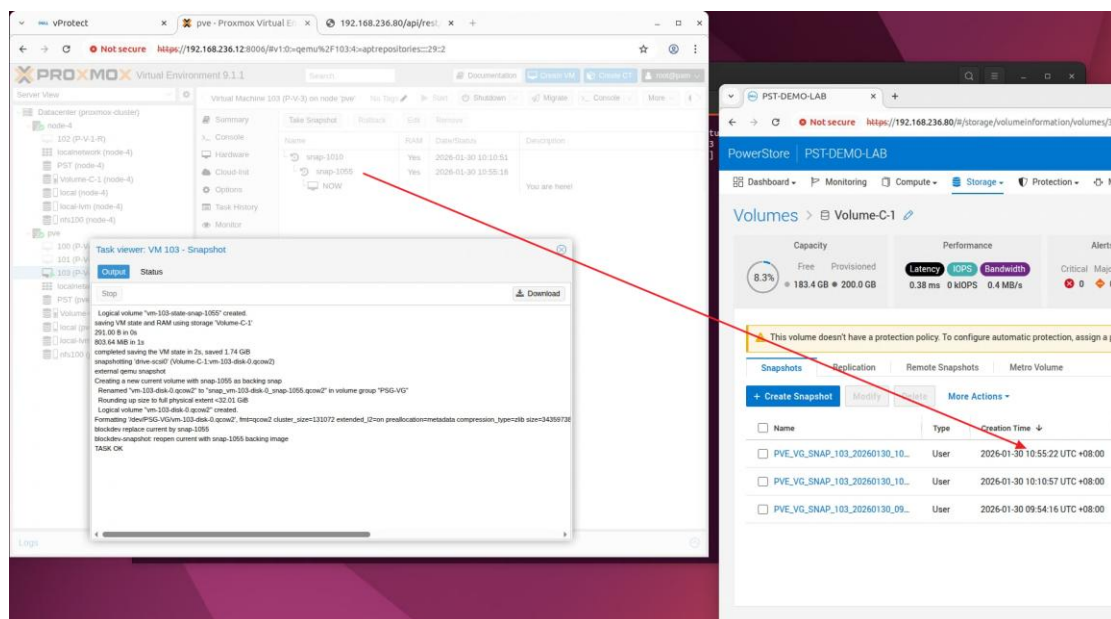
2. **事件連動**：不論是手動指令、還是網頁介面按下快照按鈕，系統都會在後台悄悄完成儲存設備端的硬體保護。
3. **自動化清理**：設定了 24 小時自動過期，避免儲存空間被快照塞爆。

## 最後的維運建議

為了讓這套機制長期穩定運行，建議您留意以下幾點：

1. **多路徑 (Multipath) 考量**：目前您的 LVM 是掛在 `/dev/sdi`。如果未來您開啟了 Multipath (為了 FC/iSCSI 冗餘)，設備名稱會變成 `/dev/mapper/mpathX`。到時候如果腳本失效，只需將 `pvs` 抓取 PV 名稱的邏輯微調即可，WWN 邏輯依然通用。
2. **監控服務健康度**：您可以偶爾檢查監控服務是否還在呼吸：`systemctl status pve-ps-monitor.service`
3. **PowerStore 端的命名與管理**：雖然我們現在是用 WWN 比對，但我建議在 PowerStore 上給這些 Volume 取個好記的名字 (例如包含 PSG-VG)，這樣萬一需要從 PowerStore 端手動還原時，您能更快速定位。

## 畫面



建構過程提問 By Gemini 分享如下

<https://gemini.google.com/share/d62d1512ee54>