



Nutanix 聖經

(AOS5.16)

作者： Steven Poitras

翻譯及審校： Nutanix 中國工程師團隊
2020.05

繁體中文化: 2024.10 台灣

最新版請參考以下英文版

<https://www.nutanixbible.com/pdf/classic.pdf>



目 錄

1 第一部分：歷史回顧	9
1.1 資料中心的演變	9
1.1.1 大型機的時代	9
1.1.2 轉向獨立的伺服器	9
1.1.3 集中式存儲	9
1.1.4 虛擬化的出現	10
1.1.5 虛擬化的成熟	10
1.1.6 固態硬碟(SSD)	11
1.1.7 雲的到來	11
1.2 延遲的重要性	12
1.2.1 頻寬的考量	13
1.2.2 記憶體延遲的影響	14
1.3 用戶空間 vs 內核空間	15
1.3.1 輪詢與中斷	16
1.3.2 移動到用戶空間/輪詢	17
1.4 Web-Scale	18
1.4.1 超融合	19
1.4.2 軟體定義的智慧化	19
1.4.3 分散式自治系統	20
1.4.4 遞增性和線性擴展	20
1.4.5 總結	20
2 第二部分：基礎	22
2.1 策略和願景	22
2.1.1 超融合/雲/虛擬化：“選擇”	22
2.1.2 Acropolis (AOS + AHV/虛擬化平臺)：“運行時”	23



2.1.3	Prism“介面”	24
2.1.4	Nutanix: 平臺	25
2.2	產品和平臺	26
2.2.1	第一步: 資料中心現代化(Core)	27
2.2.2	第二步: 建設私有雲(Essentials)	27
2.2.3	第三部: 建設混合雲(Enterprise)	28
2.2.4	平臺	29
2.3	超融合平臺	29
2.4	分散式系統	33
2.5	軟體定義	34
2.6	集群組件	35
2.7	不間斷升級	37
2.7.1	步驟 1 - 升級前檢查	38
2.7.2	步驟 2- 將升級軟體上傳到 2 節點	38
2.7.3	步驟 3 - 階段升級軟體	38
2.8	Foundation (鏡像)	39
2.8.1	架構	39
2.8.2	輸入項	40
2.8.3	系統鏡像和部署	41
2.9	磁碟機解構	46
3	第三部分 : Prism	49
3.1	設計方法	49
3.2	架構	49
3.2.1	Prism 的服務	51
3.2.2	認證和存取控制	52
3.3	管理導航	53
3.3.1	Prism Central	53
3.3.2	搜索	56



3.3.3	Prism Element	58
3.3.4	快速鍵	59
3.4	特性和使用	59
3.4.1	異常檢測	60
3.4.2	Nutanix 軟體升級	63
3.4.3	Hypervisor 升級	66
3.4.4	集群擴展(增加節點).....	69
3.4.5	I/O 度量	75
3.4.6	容量規劃	79
3.4.7	X-Play	81
3.5	APIs 介面.....	87
3.5.1	ACLI	88
3.5.2	NCLI	93
3.5.3	PowerShell CMDlets.....	96
4	第四部分：Acropolis.....	102
4.1	架構	102
4.1.1	Acropolis 服務.....	102
4.1.2	動態調度	104
4.1.3	放置位置	106
4.2	安全與加密	106
4.2.1	資料加密和金鑰管理	117
4.3	分散式存儲結構	125
4.3.1	資料結構	126
4.3.2	I/O 路徑和緩存	128
4.3.3	可擴展的中繼資料	137
4.3.4	資料保護	140
4.3.5	可用域（主機殼感知）	141
4.3.6	資料路徑冗餘	151



4.3.7	容量優化	156
4.3.8	存儲分層和優先順序	165
4.3.9	磁片平衡	167
4.3.10	快照和克隆	169
4.3.11	網路及 I/O	172
4.3.12	數據當地語系化 Data Locality	173
4.3.13	影子克隆 Shadow Clones	174
4.3.14	存儲層和監控	175
4.4	服務	177
4.4.1	Nutanix Guest Tools (NGT).....	177
4.4.2	OS 定制化.....	184
4.4.3	Karbon (容器服務)	189
4.5	備份與容災	193
4.5.1	實施組件	194
4.5.2	保護對象	195
4.5.3	備份和恢復	199
4.5.4	應用一致性快照	202
4.5.5	複製和容災 (DR)	206
4.5.6	近同步複製技術 (NearSync)	209
4.5.7	城域高可用性 Metro Availability	211
4.5.8	雲連結	214
4.6	Application Mobility Fabric*	217
4.7	管理	217
4.7.1	重要頁	217
4.7.2	集群命令	219
4.7.3	指標和閾值	225
4.7.4	Gflags.....	225
4.7.5	故障排除和高級管理	225



5 第五部分 : AHV	242
5.1 架構	242
5.1.1 節點架構	242
5.1.2 KVM 架構	242
5.1.3 最高配置和可擴展性	243
5.1.4 網路	244
5.2 如何工作	248
5.2.1 存儲 I/O 路徑	248
5.2.2 IP 地址管理	257
5.2.3 VM 高可用性 (HA)	258
5.3 管理	261
5.3.1 重要頁*	261
5.3.2 命令參考	261
5.3.3 計量與閾值	264
5.3.4 故障處理& 高級管理	264
6 第六部分 : vSphere	265
6.1 架構	265
6.1.1 節點架構	265
6.1.2 最高配置和可擴展性	266
6.1.3 網路	266
6.2 如何工作	268
6.2.1 磁碟陣列卸載負載 – VAAI	268
6.2.2 CVM Autopathing aka Ha.py	268
6.3 管理	269
6.3.1 重要頁*	269
6.3.2 命令參考	269
6.3.3 指標和閾值 *	271
6.3.4 故障排除和高級管理*	271



7	第七部分：Hyper-V.....	271
7.1	架構	271
7.1.1	節點架構	271
7.1.2	最高配置和可擴展性	272
7.1.3	網路	272
7.2	如何工作	274
7.2.1	磁碟陣列卸載負載 – ODX.....	274
7.3	管理	274
7.3.1	重要頁*.....	274
7.3.2	命令參考	274
7.3.3	指標和閾值*.....	276
7.3.4	故障排除和高級管理*.....	276
8	第八部分：Nutanix 集群.....	276
8.1	AWS 上的 Nutanix 群集.....	276
8.1.1	已支援的配置	276
8.1.2	關鍵術語 / 結構.....	277
8.1.3	架構	278
8.1.4	擺放策略	280
8.1.5	存儲	281
8.1.6	網路連接	282
8.1.7	主機網路連接	283
8.1.8	WAN / L3 網路	287
8.1.9	安全	288
8.1.10	用法和配置	288
9	第九部分：存儲服務	289
9.1	塊服務	289
9.2	檔案服務	296



9.3	物件存儲服務	301
10	第十部分：網路服務	306
10.1	Flow (微分段)	306
10.2	Epoch (網路監控/ DPI)	309
11	第十一部分：備份容災服務	309
11.1	Leap (策略驅動的災難恢復/運行手冊)	309
11.2	Mine (備份解決方案)	327
12	第十二部分：編排服務	327
12.1	Calm (編排/自動化)	327
13	第十三部分：治理服務	328
13.1	Beam (成本治理/合規性)	328
14	第十四部分：場景	328
14.1	場景：安全分析平臺	328
14.2	場景：多網站災難恢復/複製	328
15	第十五部分：集成	328
15.1	集成	328
15.1.1	OpenStack	328
16	後記	346



1 第一部分：歷史回顧

簡要回顧一下基礎設施的歷史，瞭解我們今天所處的階段。

1.1 資料中心的演變

資料中心在過去的幾十年間已經發生翻天覆地的變化，下面我們將詳細說明每一個階段。

1.1.1 大型機的時代

大型機(**MainFrame**)在相當長的一段時間內作為資料中心核心業務系統的基礎，使很多公司獲得如下主要的特徵：

- 系統本身就融合的 CPU、記憶體和存儲
- 設計為內部冗餘機制

但是大型機也帶來了如下的問題：

- 高昂的基礎架構採購費用
- 內在的複雜性
- 缺乏彈性，高度封閉的環境

1.1.2 轉向獨立的伺服器

對於大部分的公司而言，其業務很難充分利用大型機的能力，在一定程度上導致了獨立伺服器的出現，其關鍵的特性包括：

- CPU、記憶體和直連式存儲 (DAS)
- 比大型機更高靈活的系統環境
- 可通過網路訪問

但這些獨立伺服器模式仍然存在諸多問題：

- 更多孤立環境
- 資源利用率低或不均
- 對於計算和存儲而言，伺服器成為單點故障 (SPOF)

1.1.3 集中式存儲



資料是業務獲得持續盈利能力的重要一環。對於直連存儲（DAS），企業既需要更多的當地語系化存儲空間，又需要資料的高可用，以保證伺服器故障不會導致資料的不可用。

集中式存儲替代了大型機和獨立的伺服器模式，提供可共用的、更大的存儲資源池，並具有資料的保護能力，集中式存儲的主要特點包括：：

- 池化的存儲資源可以提高存儲的利用率
- 通過 RAID 的集中資料保護避免了伺服器宕機引起的資料丟失
- 通過網路進行存儲；

集中式存儲依然存在如下的問題：

- 成本高昂，但資料比硬體更有價值
- 更為複雜的系統（SAN 架構、WWPNs，RAID 組，卷，存儲控制器等）
- 需要額外的管理工具和管理團隊

1.1.4 虛擬化的出現

與此同時，我們可以觀察到，計算資源使用不充分，資源效率也低到讓人無法容忍。隨後，虛擬化被引入，使多種工作負載和作業系統和 OS 以虛擬機器(VM)的形式運行在單一物理硬體之上。虛擬化增加了業務對於物理伺服器的利用率，但也增加了豎井式環境和故障的影響範圍。其核心特徵有：

- 作業系統和物理硬體解耦合（VM）
- 高效的計算使用率可以整合工作負載

早期的虛擬化存在一定的問題：

- 豊井式環境增多，管理複雜度增加
- 缺乏虛擬機器的高可用，所以一旦計算節點失效影響範圍變得更大
- 池化資源的缺乏
- 需要單獨的管理工具和管理團隊。

1.1.5 虛擬化的成熟

成熟的虛擬化管理程式已成為高效且功能豐富的方案，隨著 vMotion（熱遷移）、HA(高可用)和 DRS(分散式資源調度)等功能的出現，用戶可以獲得提供虛擬機器高可用性和動態遷移計算工作負載的能力。但仍有一個核心問題，就是對集中式存儲的過度依賴而導致的資料路徑集中合併。這樣隨著虛擬機器快速增長和存儲陣列的負載提升會導致嚴重的存儲 IO 瓶頸。

關鍵的特性有：



- 集群化計算資源池
- 在計算節點間具備動態遷移的能力 (DRS/vMotion)
- VM 具備高可用性，防止計算節點失效
- 集中化存儲的要求

但存在以下問題：

- 虛擬機器的快速增長導致更高的存儲需求
- 存儲擴容需求導致更多豎井化環境和更高管理複雜度
- 由增加存儲導致的更高單位存儲容量成本
- 可能會導致陣列資源的爭用
- 存儲配置更加複雜：
 - ✓ Datastore 內 VM 和 LUN 的配比
 - ✓ 考慮存儲控制器的數量滿足 I/O 需求

1.1.6 固態硬碟(SSD)

SSD 可以無需借助更多的盤陣就可以實現很高的 I/O 性能，從而緩解 I/O 瓶頸。

然而，即使性能提高顯著，控制器和網路還沒有發展到能夠處理如此大量 I/O 吞吐的能力。固態硬碟的關鍵特性有：

- 比傳統 HDD 具備更好的性能
- 基本上消除了定址時間

但固態硬碟也有問題：

- 瓶頸從磁片上的存儲 I/O 上升到了存儲控制器和網路
- 豎井依然存在
- 陣列配置仍舊複雜

1.1.7 雲的到來

雲的概念是一個模糊的定義。簡單地說就是一種可以消費和利用由他人提供的服務的能力。隨著雲的引入，看待 IT、業務和最終用戶的角度已經變化。

商業組織及 IT 消費者需要 IT 提供類似雲的能力，如敏捷和快速實現價值。如果無法滿足，他們將直接採用公有雲服務，而這可能會引發 IT 新問題：如數據安全。

雲服務的核心理念：



- 自服務 / 按需消費
 - 快速實現價值(TTV) /降低門檻
- 專注服務和 SLA
 - 圍繞正常使用時間/高可靠/性能指標的合同保證
- 分級定價模式
 - 按實際使用量付費(一些服務免費)

雲的分類

多數雲的分類如下（從上層到下層）：

- 軟體即服務(SaaS)
 - 任何軟體/ 服務 通過簡單的 url 地址接入和消費
 - 例如: Workday, Salesforce.com, Google 搜索等
- 平臺即服務(PaaS)
 - 開發和部署平臺
 - 例如: Amazon Elastic Beanstalk / Relational Database Services (RDS), Google App Engine 等
- 基礎設施即服務(IaaS)
 - VMs/Containers/NFV 即服務
 - 例如: Amazon EC2/ECS, Microsoft Azure, Google Compute Engine (GCE)等

IT 焦點的轉變

雲向提出了非常有趣的 IT 兩難選擇。IT 可以擁抱它，或者尋求另一種選擇。因為他們希望把資料放在內部，但同時擁有雲的自服務和快捷的特性。這種轉變促使 IT 更像一個對公司員工提供 IT 服務的服務提供者。

1.2 延遲的重要性

下面給出特定 I/O 類型的不同延遲特性：

項目	延遲	說明
L1 cache reference	0.5 ns	
Branch Mispredict	5 ns	
L2 cache reference	7 ns	14x L1 cache
Mutex lock/unlock	25 ns	



Main memory reference	100 ns	20x L2 cache, 200x L1 cache
Compress 1KB with Zippy	3,000 ns	
Sent 1KB over 1Gbps network	10,000 ns	0.01 ms
Read 4K randomly from SSD	150,000 ns	0.15 ms
Read 1MB sequentially from memory	250,000 ns	0.25 ms
Round trip within datacenter	500,000 ns	0.5 ms
Read 1MB sequentially from SSD	1,000,000 ns	1 ms, 4x memory
Disk seek	10,000,000 ns	10 ms, 20x datacenter round trip
Read 1MB sequentially from disk	20,000,000 ns	20 ms, 80x memory, 20x SSD
Send packet CA -> Netherlands -> CA	150,000,000 ns	150 ms

(來源: Jeff Dean, <https://gist.github.com/jboner/2841832>)

上面的表格給出 CPU 訪問它的緩存需要 0.5 納秒到 7 納秒不等 (L1 vs L2) , 對於記憶體，訪問的時間為 100 紳秒，而一個固態硬碟的 4K 讀則需要 150,000 紳秒，即 0.15 毫秒。如果我們使用一個企業級固態硬碟（例如 Intel S3700 系列），這個設備能力如下：

- 隨機 I/O 性能：
 - ✓ 隨機 4K 讀可達 75000 IOPS
 - ✓ 隨機 4K 寫可達 36000 IOPS
- 序列化頻寬：
 - ✓ 持續讀可達 500 MB/s
 - ✓ 持續寫可達 460 MB/s
- 延遲：
 - ✓ 讀延遲 50 微秒
 - ✓ 寫延遲 65 微秒

1.2.1 頻寬的考量



對於傳統存儲來說，有以下幾種 I/O 媒介：

- 光纖通道 (Fiber Channel)
 - ✓ 4/8/16Gb 和 32Gb
- 乙太網(包括 FCoE)
 - ✓ 1/10Gb (40Gb Infiniband)等

我們使用 Intel S3700 系列固態硬碟的 500 MB/s 讀和 460 MB/s 寫頻寬作為計算基礎，參考如下公式：

$$\text{numSSD} = \text{ROUNDUP}((\text{numConnections} * \text{connBW} (\text{in GB/s})) / \text{ssdBW} (\text{R or W}))$$

注釋：固態硬碟數量四捨五入按照整塊計算，也不考慮 CPU 處理所有 I/O 的損耗，假設 CPU 的處理能力也是無限的。

網路頻寬		占滿頻寬所需的 SSD 數量	
控制器連接	可用網路頻寬	讀 I/O	寫 I/O
雙 4Gb FC	8Gb == 1GB	2	3
雙 8Gb FC	16Gb == 2GB	4	5
雙 16Gb FC	32Gb == 4GB	8	9
雙 32Gb FC	64Gb == 8GB	16	19
雙 1Gb ETH	2Gb == 0.25GB	1	1
雙 10Gb ETH	20Gb == 2.5GB	5	6

如上面的表格所示，如果想達到一塊固態硬碟提供的 I/O 能力理論最大值，網路可能會成為瓶頸，不同的網路頻寬對應到 1 塊到 9 塊固態硬碟的處理能力。

1.2.2 記憶體延遲的影響

典型記憶體的延遲在 100 納秒左右(不同型號及品牌的記憶體延遲會不同)，我們可以依據下面公式計算：

- 本地記憶體讀取延遲= $100\text{ns} + [\text{OS} / \text{hypervisor} \text{ 開銷}]$
- 網路記憶體讀取延遲= $100\text{ns} + \text{NW RTT latency} + [2 \times \text{OS} / \text{hypervisor} \text{ 開銷}]$



如果我們假設一個典型的網路 RTT 為 0.5 毫秒(不同交換機延遲會不同)，即 500,000 納秒，計算公式如下：

- 網路記憶體讀取延遲= $100\text{ns} + 500,000\text{ns} + [2 \times \text{OS} / \text{hypervisor 負荷}]$

如果我們假設一個非常快速的網路，RTT 只有 10,000 納秒：

- 網路記憶體讀取延遲= $100\text{ns} + 10,000\text{ns} + [2 \times \text{OS} / \text{hypervisor 負荷}]$

這就意味著即使有一個理論上非常快速的網路，與本地記憶體訪問相比，也會有超過 10,000% 的開銷，而對於一個比較慢的網路而言，這個延遲開銷相比可能超過 500,000%。

為了緩解延遲的開銷，伺服器端的緩存技術便應運而生了。

1.3 用戶空間 vs 內核空間

在內核空間還是在使用者空間處理內容的爭論是一個經常被辯論的話題。在這裡，我將解釋每一部分內容和他們各自的優點與缺點。

任何作業系統 (OS) 都有兩個核心執行區域：

- 內核空間
 - 作業系統中最有特權的部分
 - 處理調度，記憶體管理等
 - 包含物理設備驅動程式並處理硬體交互
- 用戶空間
 - “其他的一切”
 - 這是大多數應用程式和進程所在的地方
 - 受保護的記憶體和執行

這兩個空間協同工作，以使作業系統運轉，在繼續講述用戶空間和內核空間之前，我們來先定義一些關鍵元件：

- 系統調用
 - 又名 內核調用 (kernel call)，通過中斷（稍後會在後面描述）從內核完成某個活動進程的請求
- 上下文切換
 - 將執行從進程轉移到內核，反之亦然

例如，下面是一個簡單的將輸入寫入磁片的應用的用例，接下來會發生：

1. 應用程式想要將資料寫入磁片
2. 請求一個系統調用

3. 上下文切換到內核
4. 內核覆制數據
5. 通過驅動程式執行寫入磁片

下面為交互過程示意圖：

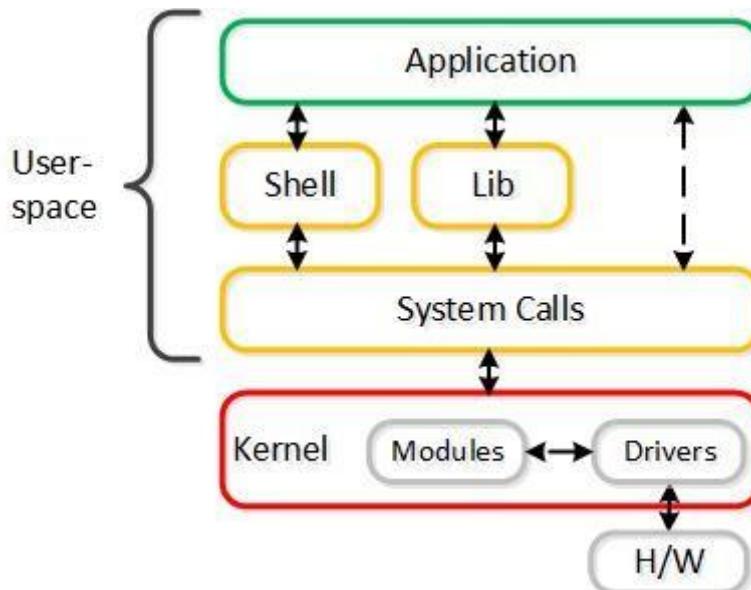


圖.用戶和內核空間交互示意

其中一個比另一個好嗎？實際上用戶空間和內核空間各有優點和缺點：

- 用戶空間
 - 非常靈活
 - 孤立的故障域（進程）
 - 可能效率低下
 - 上下文切換花費時間（~1,000ns）
- 內核空間
 - 非常僵化。較大故障域
 - 可能更高效率
 - 減少上下文切換

1.3.1 輪詢與中斷

另一個核心部分是如何處理兩者之間的交互。有兩種主要的交互類型：

- 輪詢
 - 不斷“輪詢”，例如 一直發出需求
 - 例子：滑鼠，顯示器刷新率等



- 需要持續的 CPU 處理，但延遲較低（但是極大地減少延遲）
- 消除內核中斷處理常式的開銷
 - 刪除上下文切換
- 中斷
 - “對不起，我需要一個 `foo` 請求”
 - 舉例：舉手請求資源
 - 可能使“CPU 高效”，但要視場景而定。
 - 通常會有更高的延遲

1.3.2 移動到用戶空間/輪詢

隨著設備速度變得越來越快（例如 NVMe，Intel Optane，pMEM），內核和設備之間的交互已經成為瓶頸。為了消除這些瓶頸，許多供應商通過輪詢將內容從內核移動到用戶空間，以得到更好的結果。

英特爾存儲性能開發套件（SPDK）和資料平面開發套件（DPDK）就是很好的例子。這些項目旨在最大限度地提高性能並盡可能減少延遲，並取得了巨大的成功。

這種轉變包括兩個關鍵的內容：（這種轉變由兩個核心改變組成）

1. 將設備驅動程式移動到使用者空間（原先是內核空間）
2. 使用輪詢方式（原先是中斷方式）

與之前基於內核的設備驅動方式相比，這可以實現更高的性能，因為它消除了：

- 代價高昂的系統調用和中斷處理
- 數據副本
- 上下文切換

下面為使用者空間中驅動程式與設備的交互示意圖：

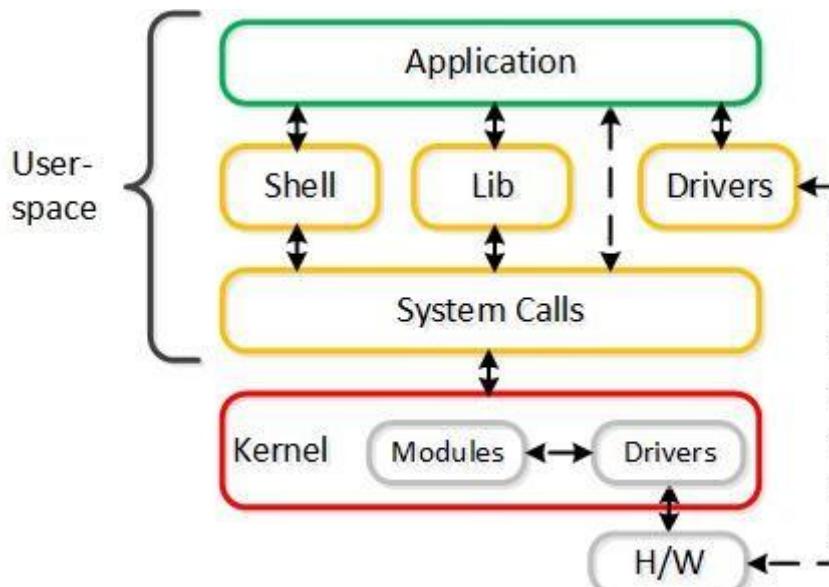


圖.用戶空間和輪詢交互

實際上，Nutanix 為其 AHV 產品（vhost-user-scsi）開發的一款軟體實際上正在被英特爾用於他們的 SPDK 項目。

1.4 Web-Scale

名詞解釋：**Web Scale**，一種全新的計算架構方式，面向基礎架構和計算資源等，可以實現互聯網規模的彈性擴展能力。

這部分將介紹一些 **Web Scale** 基礎架構的核心概念，以及為什麼我們要採用它。開始介紹前，必須清楚的瞭解 **Web Scale** 不意味著你真的要擴展到像 Google、Facebook 或 Microsoft 等這種互聯網規模。這種架構適用於任何擴展情況(從 3 節點到上千節點)，並且會受益匪淺。

現有的挑戰包括：

- 複雜，複雜，還是複雜
- 業務快速增長的需要
- 對敏捷性的需求等

當討論 **Web Scale** 架構時，有如下一些關鍵概念：

- 超融合
- 軟體定義的智慧化
- 分散式自治系統
- 系統的遞增性和線性擴展能力



其他相關概念：

- 基於 API 的自動化和豐富的分析能力
- 作為核心租戶的安全
- 自恢復能力

下面從技術視角來解釋這些概念的意義。

1.4.1 超融合

在業界，針對於超融合到底是什麼有著很多不同的看法。這也是因為元件（虛擬化、網路等）範圍的不同而產生的差異。但是，核心的概念是這樣闡述：天然地（**Natively**）將兩個或多個元件組合到一個獨立的單元中。在這裡，天然(**Natively**)是一個關鍵字。為了更加有效率，元件一定是天然地整合在一起，而不是簡單地捆綁在一起。對於 Nutanix，我們天然地將計算和存儲融合到我們設備的單一節點中。這就真正意味著天然地將兩個或多個元件融合在一個獨立的、可容易擴展的單元中。

優勢在於：

- 獨立單元的擴展
- 本地 I/O 處理
- 消除傳統計算/存儲的豎井式結構，將其融合在一起

1.4.2 軟體定義的智慧化

軟體定義的智慧化是在通用的、商品化的硬體之上通過運行軟體來實現核心的邏輯，而這些邏輯之前用專有的硬體程式設計方式實現(例如 ASIC/FPGA 等)。對於 Nutanix而言，我們將傳統的存儲邏輯(例如 RAID，去重，壓縮等)採用軟體方式去實現，這些軟體運行在標準的 x86 硬體上的 Nutanix 控制虛擬機器 CVM(Controller Virtual Machine)內。（注：Nutanix 支援 x86 和 IBM POWER 架構）那就真正意味著把關鍵處理邏輯從專有硬體中剝離，並在商用的硬體上進行軟體操作。

好處在於：

- 快速地版本反覆運算週期
- 消除了專有硬體的依賴
- 利用通用商業硬體提高經濟性
- 使用期限內的投資保護

最後一點需要特別說明一下，投資保護是指舊的硬體可以運行最新的和最好的軟體。這意味著，在其貶值週期內的一部分硬體依然可以運行最新的軟體，並且可以與新部署的設備具有相一致的特性。



1.4.3 分散式自治系統

分散式自治系統涉及從傳統的單一集線模式處理業務轉向跨集群內的所有節點分散式處理業務，可以考慮創建一個完全分散式的系統。傳統角度考慮問題是假設硬體是可靠的，在某種程度上是對的。然而分散式系統的核心思想是硬體終究會出問題，用一個優雅的、業務不間斷的方式中處理故障是關鍵點。這些分散式系統的設計是為了調整和修復故障，達到自恢復和自治的目的。在元件發生故障時，系統將透明地處理和修復故障，並持續按照預期運行。將會提醒使用者知曉故障的存在，但不會作為一個緊急事件被提出來，任何一種修復(如：替代一個失效的節點)都可以按照管理員事先設定好的計畫表去自動化的處理。另外一種方式是重建而不需要替換，在 **master** 失敗的情況下，會選出一個新的 **master**，利用 **MapReduce** 的機制來分配任務的處理。其真正意義在於：

- 將角色和職責分配給系統中的所有節點
- 利用 **MapReduce** 等機制執行分散式任務處理
- 當需要一個新的主資料節點時，採用選舉機制

優勢：

- 消除單點故障 (SPOF)
- 分散式業務負載，消除任何瓶頸

1.4.4 遞增性和線性擴展

遞增性和線性擴展是指現代化系統必須具有的兩種能力，前者是指可以從一組小規模資源構建，後者是當指按需擴展的時候可以線性增加系統的性能。上面提到的所有結構都是使得線性擴展成為可能的關鍵因素。例如，一個運行虛擬化負載的傳統三層架構：伺服器、存儲和網路，所有這些元件都是獨立擴展的。當擴展伺服器數量時，卻不能實現存儲性能的橫向擴展。但是像 Nutanix 的超融合平臺，卻可以同時隨著節點數的增加而同時擴展：

- 計算節點數量
- 存儲控制器的數量
- 計算和存儲的性能及容量
- 參與集群運行的節點數量

這就意味著：

- 存儲和計算節點的線性增加可以實現性能和容量的線性增長

優勢在於：

- 從小規模開始擴展；
- 在任何規模下獲得一致性的性能增長

1.4.5 總結



綜上所述：

- 計算資源利用率的低效導致向虛擬化轉移
- 高級特性，包括 vMotion、HA 和 DRS 等需要集中式存儲
- VM 的快速增長不僅增加存儲負載，而且增加存儲資源爭用，帶來瓶頸
- 固態硬碟緩解了磁片 I/O 問題，但依然不能解決網路和控制器帶來的瓶頸
- 跨網路的緩存或記憶體訪問將面臨巨大的開銷，優勢不再明顯
- 陣列配置複雜性依然存在
- 伺服器端的緩存可降低陣列的負載和網路影響，但是必須引入新的元件
- 當地語系化有利於降低傳統網路傳輸資料（跨節點讀）所面臨的瓶頸和開銷
- 將焦點從複雜的基礎架構轉到管理的易用性和系統堆疊的簡化
- Web Scale 的世界誕生了



2 第二部分：基礎

2.1 策略和願景

當我們在構思 Nutanix 這個公司時就認為，它應該專注於一個目標：

在任意位置讓基礎架構計算資源更為透明。

這種簡單性將通過集中注意三個核心領域來實現：

1. 提高選擇和可攜性(HCI/Cloud/Hypervisor)
2. 通過聚合、抽象和智慧軟體簡化“棧”(Acropolis)
3. 通過關注用戶體驗(UX)和設計(Prism)，提供直觀的使用者介面(UI)

2.1.1 超融合/雲/虛擬化：“選擇”

儘管我們從支援單個虛擬化系統管理程式(ESXi)的單一硬體平臺(NX)開始，但我們始終知道我們不僅僅是一個系統管理程式/平臺/雲公司。這是我們選擇從頭構建自己的 UI 的原因之一，而不是在 vCenter 中作為外掛程式運行，也不是在內核中作為 VM 運行(還有更多的原因)，等等。你可能會問為什麼？選擇。

沒有一個虛擬化系統管理程式、平臺或雲可以滿足所有客戶的需求。通過支援同一平臺下的多種解決方案，我們為客戶提供了選擇和影響力。通過賦予它們在它們之間移動的能力，我們賦予了它們靈活性。所有交付與相同的經驗，因為它都是 Nutanix 平臺的一部分。

我們現在支援超過 12 種不同的硬體平臺(直接/OEM/協力廠商)、多個虛擬化管理程式(AHV、ESXi、Hyper-V 等)，並擴展與所有主要雲供應商(AWS、Azure、GCP)的集成。這允許客戶選擇對他們最有利的方式，並將其用於供應商之間的協商。

注意：平臺是一個關鍵字，在整個章節和一般情況下都會用到。我們不是在嘗試打造一次性的產品，我們是在打造一個平臺。

下面展示了 Nutanix 平臺的高級架構：

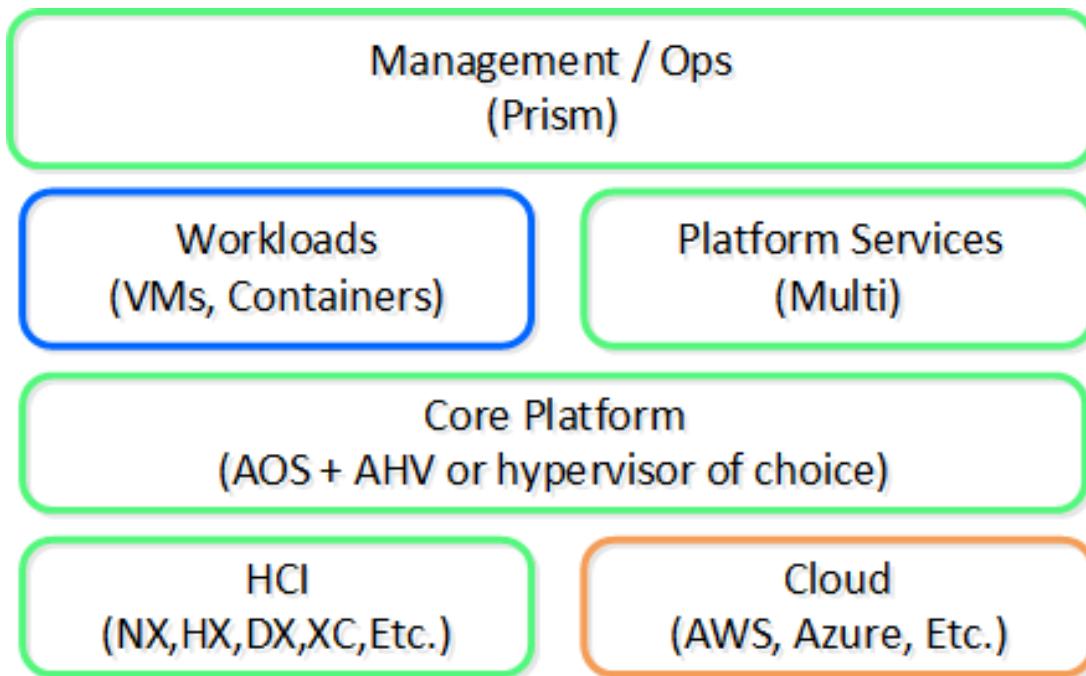


圖. Nutanix 平臺-架構

2.1.2 Acropolis (AOS + AHV/虛擬化平臺):“運行時”

我們通過一種稱為分散式存儲結構(DSF 當時被稱為 Nutanix 分散式檔案系統，即 NDFS)的特性簡化了存儲，將本機存放區資源與智慧軟體結合起來，提供類似“集中式存儲”的功能，從而開始構建我們的全新的系統。

多年來，我們添加了大量的特性和功能。為了簡化問題，我將其分為兩個核心領域：

1. 核心服務 -基礎服務
2. 平臺服務 - 以提供額外功能/服務的核心服務為基礎的服務

核心提供基本的服務和元件，這些服務和元件有助於工作負載(vm /容器)和其他更高級別的 Nutanix 服務的運行。一開始這只是 DSF 產品，但是我們繼續擴展平臺的功能來說明簡化和抽象堆疊。



下面是 AOS 核心平臺的高層視圖:

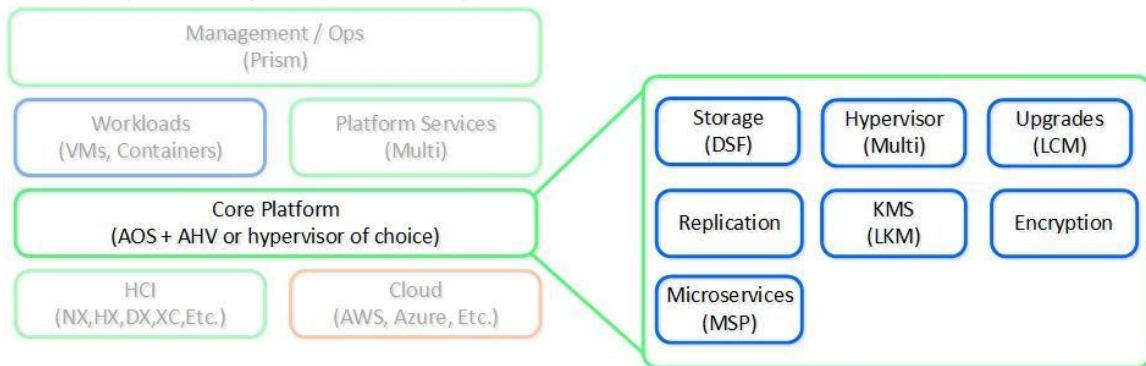


圖. Nutanix 平臺-Acropolis 核心

多年來，通過引入我們自己的虛擬化系統管理程式(AHV)、簡化升級和提供安全性和加密等其他基本服務，這已經擴展到抽象虛擬化之類的東西(我們認為這應該是透明的，並且是系統的一部分)。

有了這些功能，我們解決了許多基礎設施級別的問題，但我們並沒有止步於此。人們仍然需要額外的服務，如檔共用、物件存儲或容器。

而不是要求客戶使用其他供應商和產品的一些服務，我們認為哪些我們應該合作，哪些我們應該建立自己的。在備份方面，我們與 Veeam 和 Hycu 等供應商合作。在檔和物件服務方面，我們將它們作為服務構建到平臺中。

下面是 Nutanix 平臺服務的高級視圖:

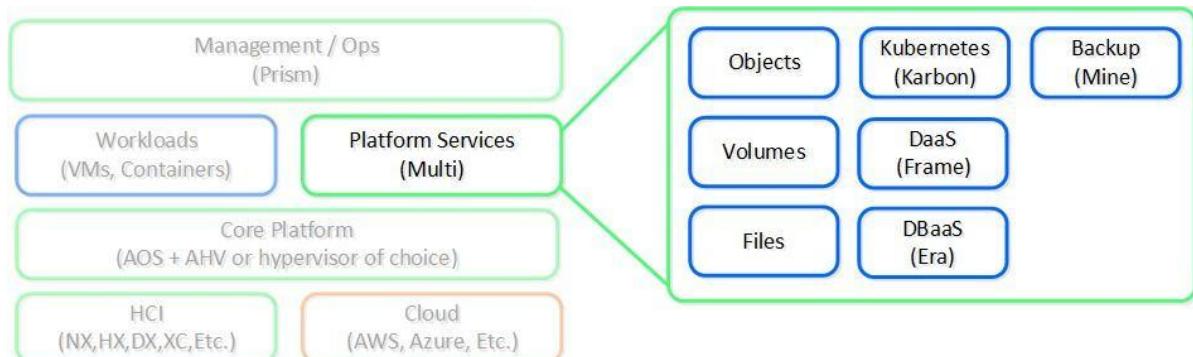


圖. Nutanix 平臺- 服務

2.1.3 Prism“介面”

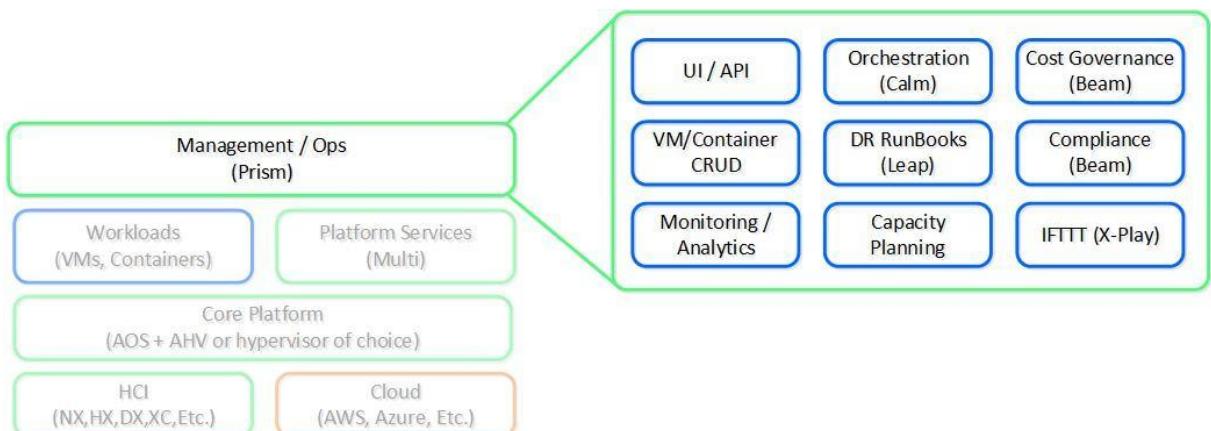


圖. Nutanix 平臺- Prism

簡單地說，應用一些像蘋果這樣的公司所提倡的設計原則，它們注重簡單、一致和直覺。從一開始，我們就在 Nutanix 產品的“前端”上投入了大量的時間和精力。UI/UX 和設計團隊並不是事後才想到的，而是一直在推動其發展。舉個例子，我們是第一批將管理介面用 HTML5 編寫的企業軟體公司(除了 SaaS 公司)之一。

這裡的另一個核心項目是專注於為平臺提供一個單一的介面，並在整個過程中保持一致的體驗。我們的目標是聚合 UI，就像融合基礎設施一樣。我們希望 Prism 成為一個單一介面，允許您管理和使用 Nutanix 平臺，無論是管理資料中心中的虛擬化、雲中的桌面即服務，還是提供開銷可視性。

這對於我們繼續通過功能/服務的創建和獲取來擴展平臺是很重要的。與其將新功能捆綁在一起，我們寧願花時間將它們原生地集成到平臺中。這是一個較慢的過程，但從長遠來看，它可以保持經驗的一致性並降低風險。

2.1.4 Nutanix: 平臺

總而言之，我們的願景很簡單：“一個平臺，任何應用，任何地點”。

注意:

我從市場部的術語中借鑒了這句話，它非常適合並且簡明扼要地說明了我們的目標。

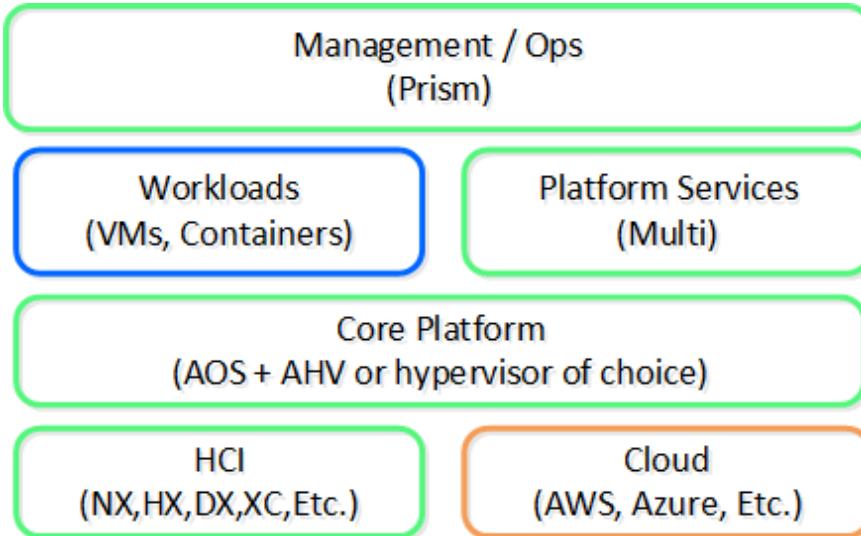


圖. Nutanix 平臺- 架構

從一開始，這就是我們的目標。這是我在 2014 年創建的一個關於 Nutanix 平臺架構的圖片。正如你所看到的，沒有太多的變化，我們只是繼續擴大和朝著這個目標努力。

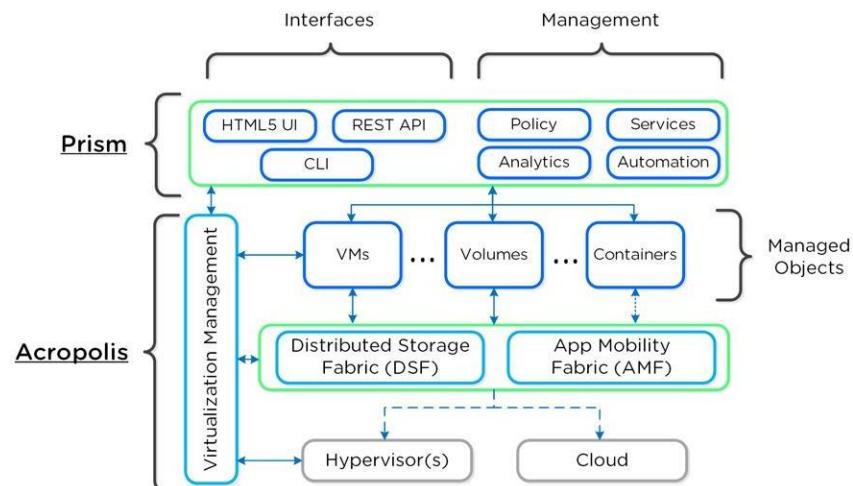


圖. Nutanix 平臺- Circa 2014

2.2 產品和平臺

多年來，Nutanix 平臺的能力和服務都有了很大的增長。多年來，它已經發展到簡化和抽象虛擬化、自動化升級和操作以及更多。本節將介紹當前的產品組合和夥伴關係。注:請參閱 Nutanix 網站的最新投資組合和產品。



多年來，隨著產品系列的豐富，我們更專注於結果以及實現這些目標的過程，而不僅是談論產品。以下步驟涵蓋了客戶的“旅程”以及 Nutanix 可以幫助他們實現的結果。

2.2.1 第一步：資料中心現代化(Core)

Core 包括基本的 Nutanix 產品，可促進從複雜的 3 層基礎架構遷移到簡單的 HCI 平臺。AOS 提供所有核心服務（存儲，升級，複製等），Prism 提供控制平面和管理主控台，AHV 提供免費的虛擬化平臺（注意：您也可以使用 ESXi，Hyper-V 和 XenServer）。

Core 功能包括：

- 核心平臺(HCI)
- 存儲服務
- 虛擬化
- 集中管理與運營
- 升級
- 複製 / DR

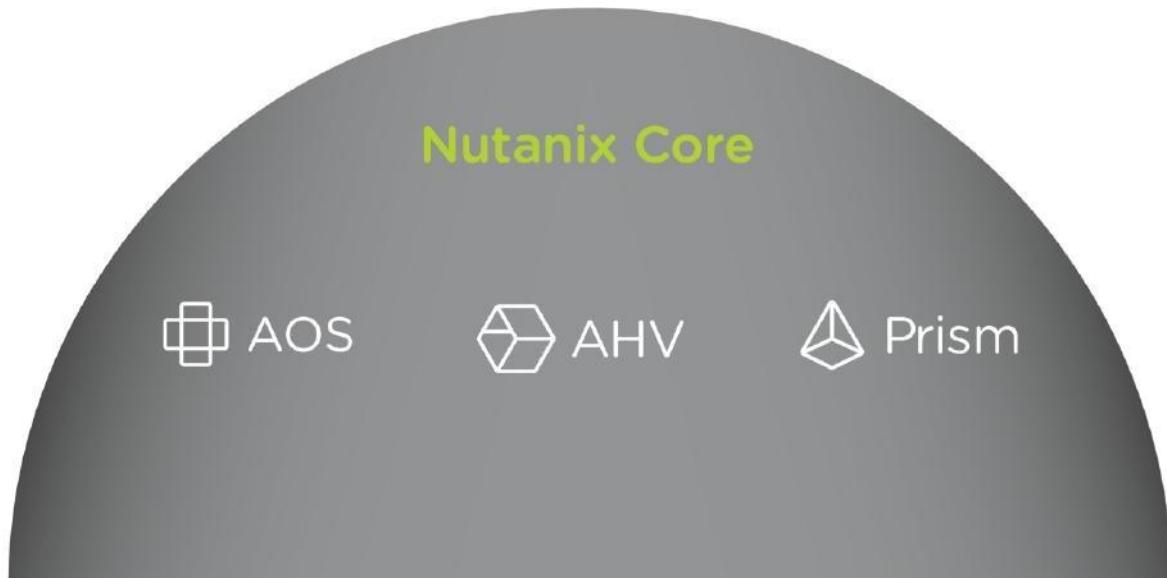


圖. Products Ecosystem – Core

2.2.2 第二步：建設私有雲(Essentials)



Essentials 專注於提供如私有雲一樣的消費 Core 基礎架構的功能。 **Flow** 提供網路分段和安全性，**Files** 提供檔服務，**Calm** 提供自助服務，配額和編排功能。

基本功能包括：

- 高級分析和異常檢測
- 自動化與編排
- 自助服務門戶（SSP）和配額
- 微分段
- 檔服務

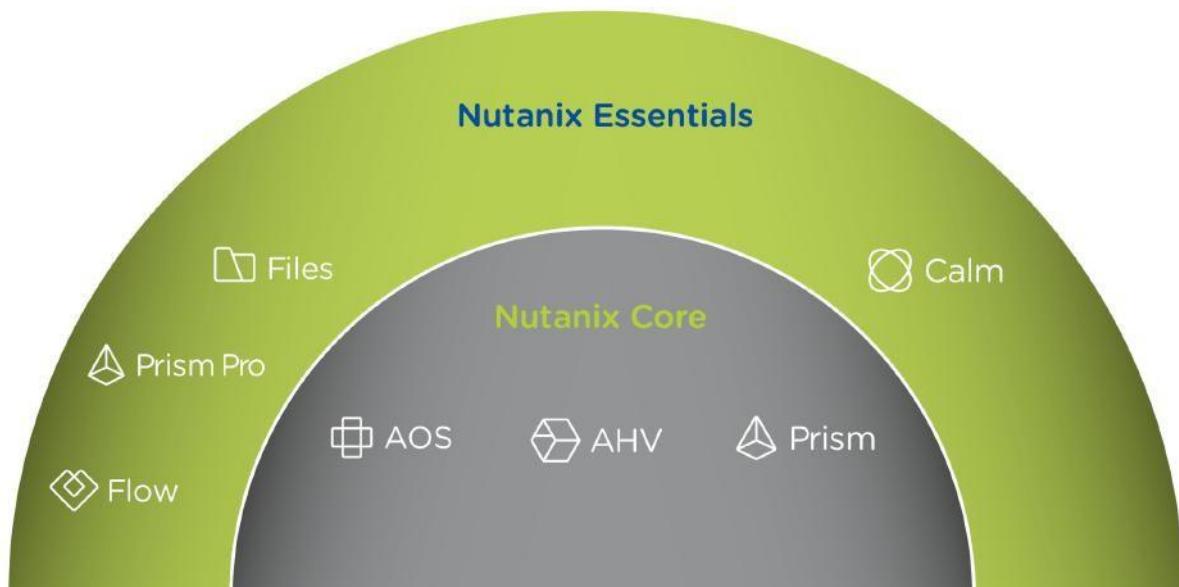


圖. Products Ecosystem - Private Cloud

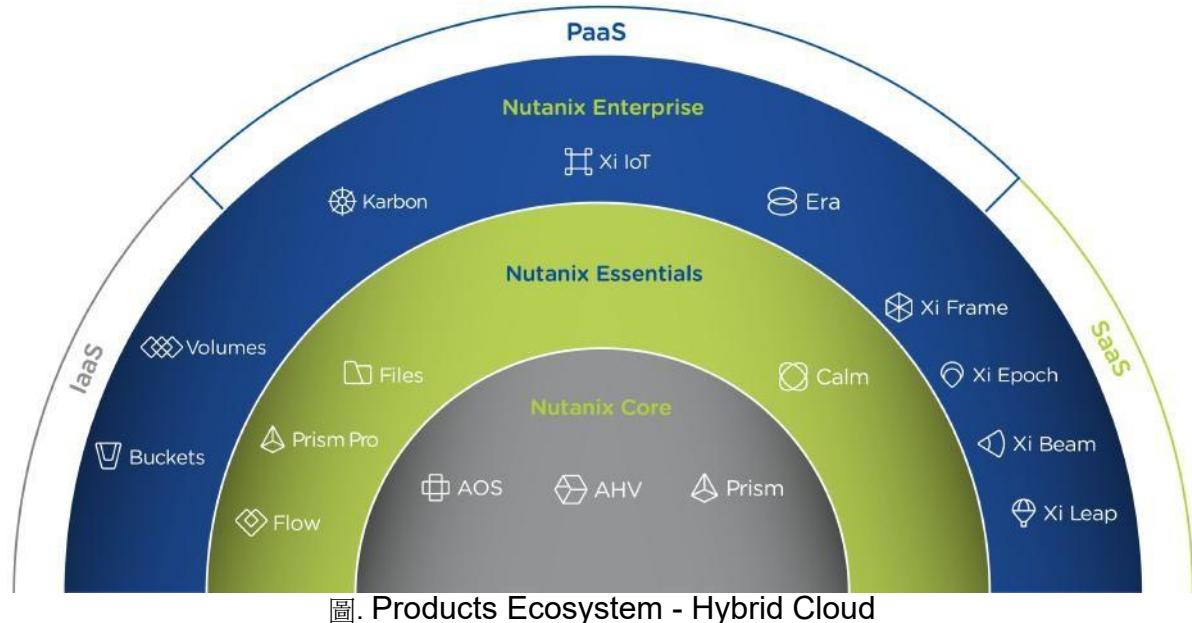
2.2.3 第三部：建設混合雲(Enterprise)

Enterprise 專注於提供在雲和雲服務之間遷移工作負載的能力。其中包括 **Beam** 等功能，這些功能側重於跨雲和本地部署的成本管理和合規性，以及 **Frame (DaaS)** 和 **Xi Leap (DRaaS)** 等其他雲服務。

Enterprise 功能包括：

- 策略驅動的災難恢復/運行手冊自動化
- DRaaS
- 混合雲成本治理和合規性
- 桌面即服務 (DaaS)
- 資料庫即服務 (DBaaS)
- Kubernetes / Docker 服務

- 物件存儲
- 塊服務



2.2.4 平臺

Nutanix 目前支持如下平臺：

- Nutanix 設備
 - NX (Supermicro)
 - HPE Powered DX
- OEM 設備
 - Nutanix on Lenovo HX
 - Nutanix on IBM CS
 - Nutanix on Dell XC
- 協力廠商伺服器支援
 - Nutanix on HPE ProLiant and Apollo
 - Nutanix on Cisco UCS
 - Nutanix on Intel Data Center Blocks
 - Nutanix Tactical and Ruggedized platforms on Klas

2.3 超融合平臺

您可以觀看以下講解的視頻: [LINK](#)

超融合系統有一些核心結構：

- 必須融合整合計算堆疊（例如，計算+存儲）
- 必須在系統中的各個節點之間分佈（分發）資料和服務
- 必須呈現和提供與集中存儲相同的功能（例如，HA，即時遷移等）
- 必須使資料盡可能靠近執行（計算）（Importance of Latency）
- 不依賴於虛擬機器管理程式
- 不依賴於硬體

下圖顯示了典型的三層堆疊與超融合的示例：

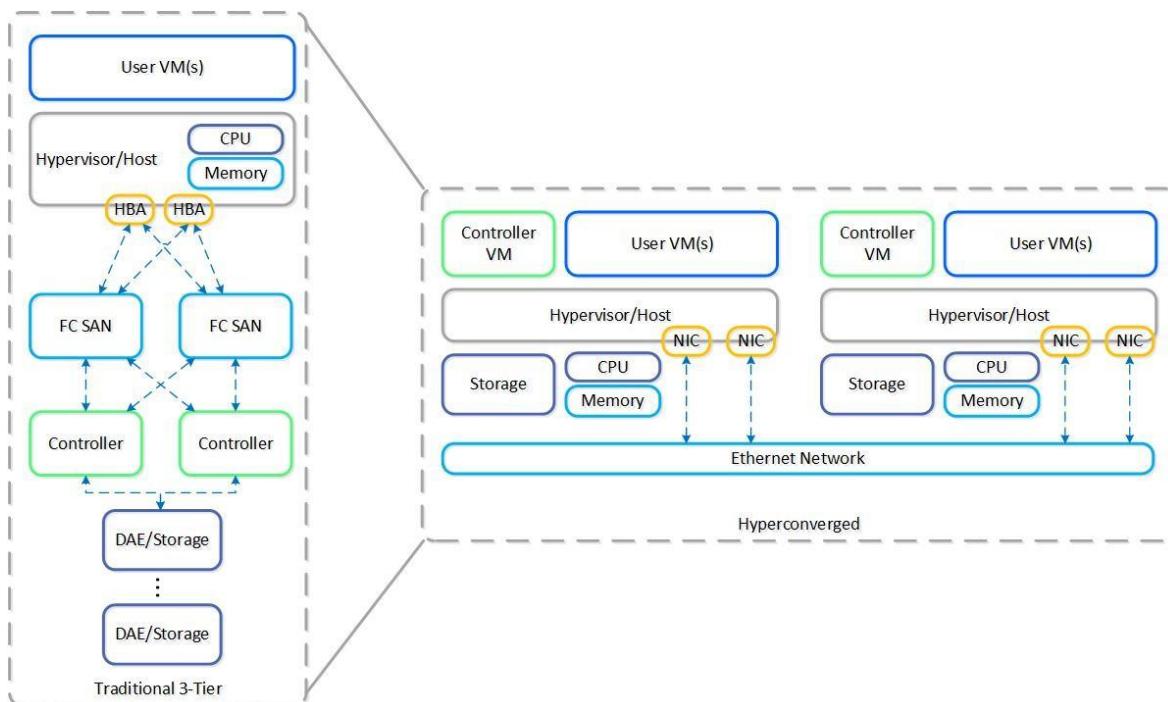


圖. 3-Tier vs. HCI

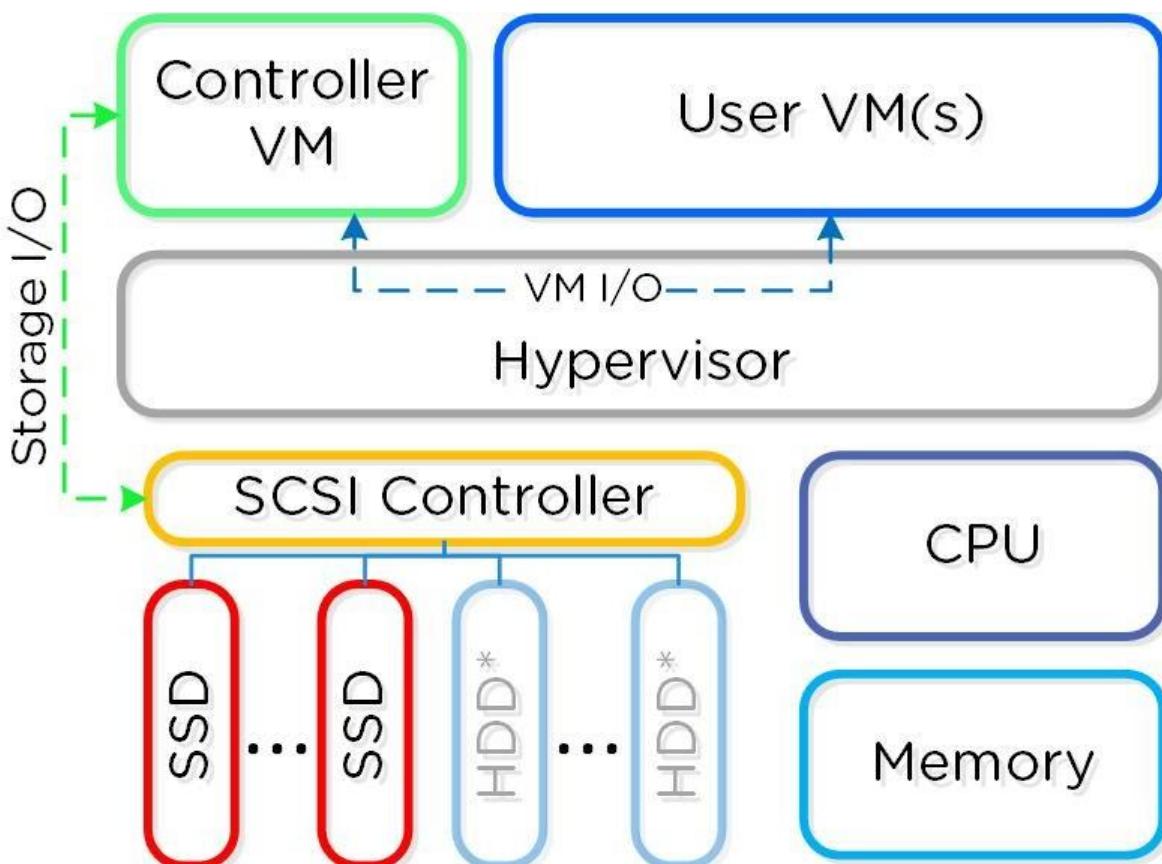
如您所見，超融合系統執行以下操作：

- 虛擬化並將控制面移動到主機
- 通過軟體提供核心服務和邏輯
- 在系統中的所有節點之間分佈（分片）資料
- 將存儲移動到計算本地

Nutanix 解決方案是一個融合了存儲和計算資源於一體的解決方案。它利用本機群組件來為工作負載構建一個分散式的平臺。

每個節點運行一個業界標準的 hypervisor (ESXi, AHV, Hyper-V 和 XenServer) 和 Nutanix 控制器虛擬機器 (CVM) 。 Nutanix CVM 中運行著 Nutanix 核心軟體，服務於主機上虛擬化層和虛擬機器相關的所有 I/O 操作。

下圖提供了一個例子，解釋了典型的節點邏輯架構：



*All flash nodes will only have SSD devices

圖 10-3. Converged Platform

Nutanix CVM 負責核心 Nutanix 平臺邏輯，並處理以下服務：

- 存儲 I / O 和轉換（重複資料刪除，壓縮，EC）
- UI / API
- 升級
- DR /複製
- 其他



注意：某些服務/功能將生成其他服務輔助 VM 或使用微服務平臺（MSP）。例如，Nutanix Files 將部署其他 VM，而 Nutanix Objects 將為 MSP 部署 VM 並利用它們。

對於運行 VMware vSphere 的 Nutanix 節點，管理 SSD 和 HDD 設備的 SCSI 控制器直接通過 VM-Direct 路徑(Intel VT-d)傳遞給 CVM。在 Hyper-V 的情況下，存放裝置通過 passed through 傳遞到 CVM。

虛擬化控制器

在用戶空間中將 Nutanix 控制器作為 VM 運行的關鍵原因實際上歸結為兩個核心領域：

1. 流動性
2. 彈性
3. 維護/升級
4. 性能，真的

從一開始，我們就知道我們不僅僅是一家平臺公司。從這個意義上講，無論對於硬體，雲還是虛擬機器管理程式供應商，選擇對於我們來說一直都是一件大事。

通過在用戶空間中作為 VM 運行，它將 Nutanix 軟體與底層虛擬機器管理程式和硬體平臺分離。這使我們能夠快速添加對其他虛擬機器管理程式的支援，同時在所有操作環境（本地和雲）中保持核心代碼庫不變。此外，它為我們提供了不受限於特定于供應商的發佈週期的靈活性。

由於在用戶空間中作為 VM 運行的性質，我們可以優雅地處理諸如升級或 CVM“故障”之類的事務，因為它們不在虛擬機器管理程式之內。例如，如果某個災難性問題導致 CVM 崩潰，則整個節點仍將繼續使用來自群集中其他 CVM 的存儲 I/O 和服務進行操作。在 AOS（Nutanix 核心軟體）升級期間，我們可以重新啟動 CVM，而不會對該主機上運行的工作負載產生任何影響。

但是，進入內核不是快得多嗎？簡單的答案，不。

一個常見的討論話題是圍繞內核還是用戶空間的爭論。作為背景問題，我建議閱讀“用戶與內核空間”一節，其中涵蓋了兩者的實際含義和優點和缺點。

總而言之，作業系統（OS）有兩個執行區域：內核（驅動程式可能位於作業系統的特權內核）和用戶空間（應用程式/進程位於其中）。傳統觀點認為，就用戶空間和內核（又稱為上下文交換 context switches）之間的移動而言，在 CPU 和時間（~1,000ns /上下文切換）方面開銷高昂。

爭論觀點認為處於內核中總是更好/更快。事實並非如此。因為無論如何，在 guest VM 的 OS 中一定會存在上下文交換 context switches。

2.4 分散式系統

分散式系統有三個核心構成：

- 必須沒有單點故障(SPOF)
- 在任何規模上都不能有任何瓶頸(必須是線性擴展的)
- 必須利用併發(MapReduce)

總之，一組 Nutanix 節點形成了一個分散式系統（Nutanix 集群），負責提供 Prism 和 Acropolis 功能。所有服務和元件都分佈在集群中的所有 CVM 中，以提供大規模的高可用性和線性性能。

下圖展示了這些 Nutanix 節點是如何形成一個 Nutanix 集群的：

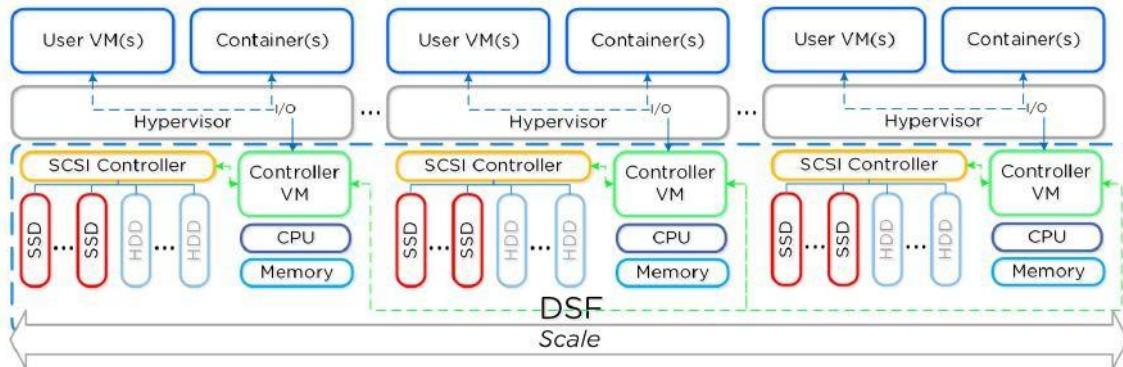


圖 .Nutanix 集群-分散式系統

這些技術也適用於中繼資料和資料。通過確保中繼資料和資料分佈在所有節點和所有磁片設備上，可以在正常的資料登錄和重新保護期間保持最高的性能。

這使我們的 MapReduce 框架（Curator）能夠充分利用集群的全部能力，並存執行操作。這些操作包括資料重新保護，壓縮，糾刪碼，重複資料刪除等。

下圖顯示了每個節點處理的工作負載如何隨著集群的擴展而急劇下降的%：

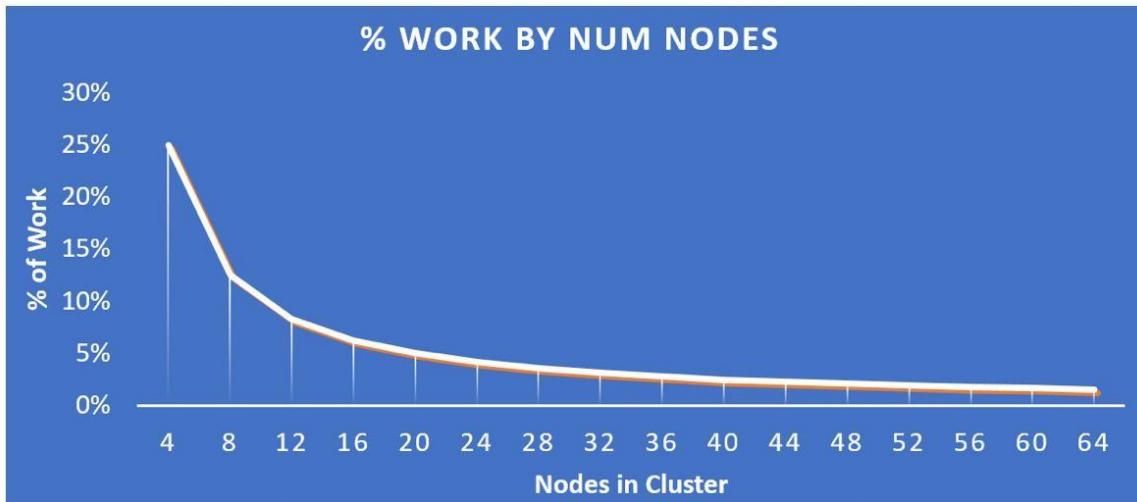


圖 . 工作分配-集群擴展

關鍵點:隨著集群中節點數量的增加(集群擴展)，某些活動實際上變得更加高效，因為每個節點只處理一部分工作。

2.5 軟體定義

軟體定義系統有四個核心構成:

- 必須提供平臺移動性(硬體、Hypervisor)
- 不能依賴任何特定硬體
- 必須支援快速的開發速度(特性、bug 修復、安全補丁)
- 必須利用摩爾定律

正如之前所提到的，Nutanix 平臺是一個基於軟體的而以軟+硬體設備捆綁方式交付的解決方案。控制器虛擬機器（CVM）中實現了 Nutanix 軟體絕大多數功能與邏輯，從一開始就被設計成可擴展和可插入的體系架構。軟體定義而非依賴於硬體卸載或構造的一個關鍵優勢在於可擴展性。與任何產品生命週期一樣，將始終引入改進和新特性。

由於不依賴於任何定制化的 ASIC/FPGA 或硬體功能，Nutanix 以軟體更新的方式即可開發和部署新功能。這意味著可以通過軟體升級來獲得新功能（如：重複資料刪除）。這也讓那些老型號的設備可以支援新型號上推出的功能。比如說，你正在一款老的機型上（例如：2400）運行著工作負載，而它沒有提供資料去重的功能。你覺得能從這項功能中大大受益，因此就在不中斷業務的情況下進行線上升級。馬上你就有了這項資料去重的能力，如此簡單而已。

就像增加新的軟體功能，你也可以為 DSF 創建新的適配器或介面。當產品剛出廠時，它只從虛擬化層支持 I/O iSCSI，現在已經擴展到包括 NFS 和 SMB。將來我們

支援為不同工作負載和 hypervisor (HDFS 等) 創建新的適配器。再次強調，所有這些都能通過軟體升級就能實現。這跟傳統架構的獲得“最新最好功能”就一定要硬體更新或另外購買軟體形成極大反差。Nutanix 把它變得不一樣，由於所有功能都通過軟體部署，所以它能運行在任何硬體平臺和任何 hypervisor，並通過軟體升級實現新的部署。

下圖是一個對於軟體定義的控制框架的邏輯展現：

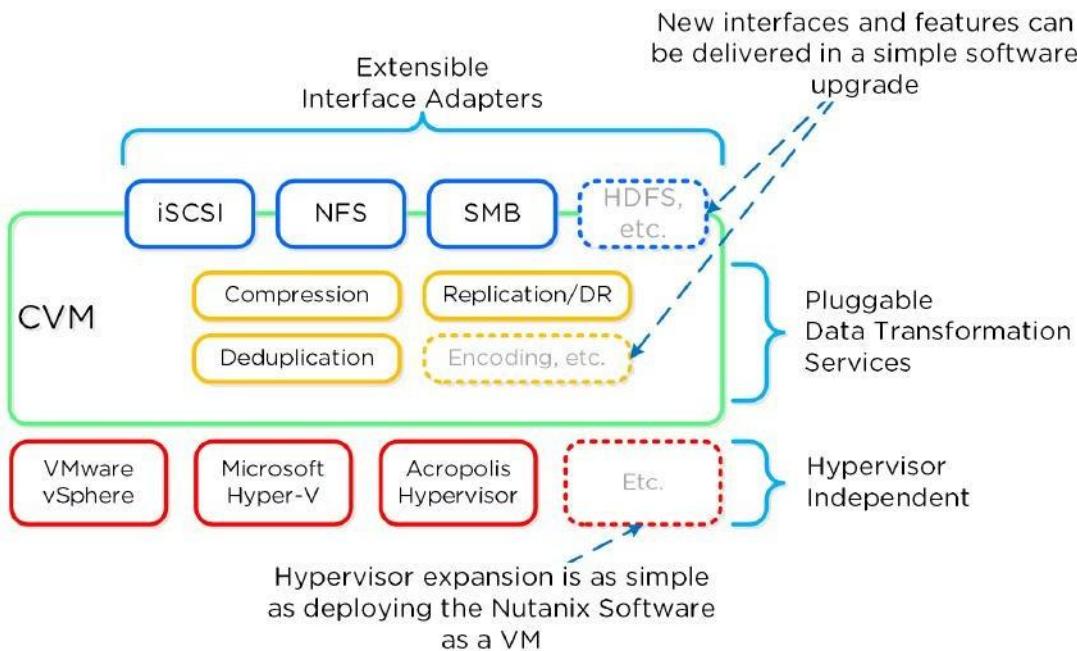


圖. 軟體定義的控制器框架

2.6 集群組件

你可以通過觀看下面視頻來說明理解：https://youtu.be/3v5RI_lbfV4

面向使用者的 Nutanix 產品的部署和使用非常簡單。這主要是通過抽象和軟體中的大量自動化/集成來實現的。

以下是 Nutanix Cluster 主要組件的詳細視圖：

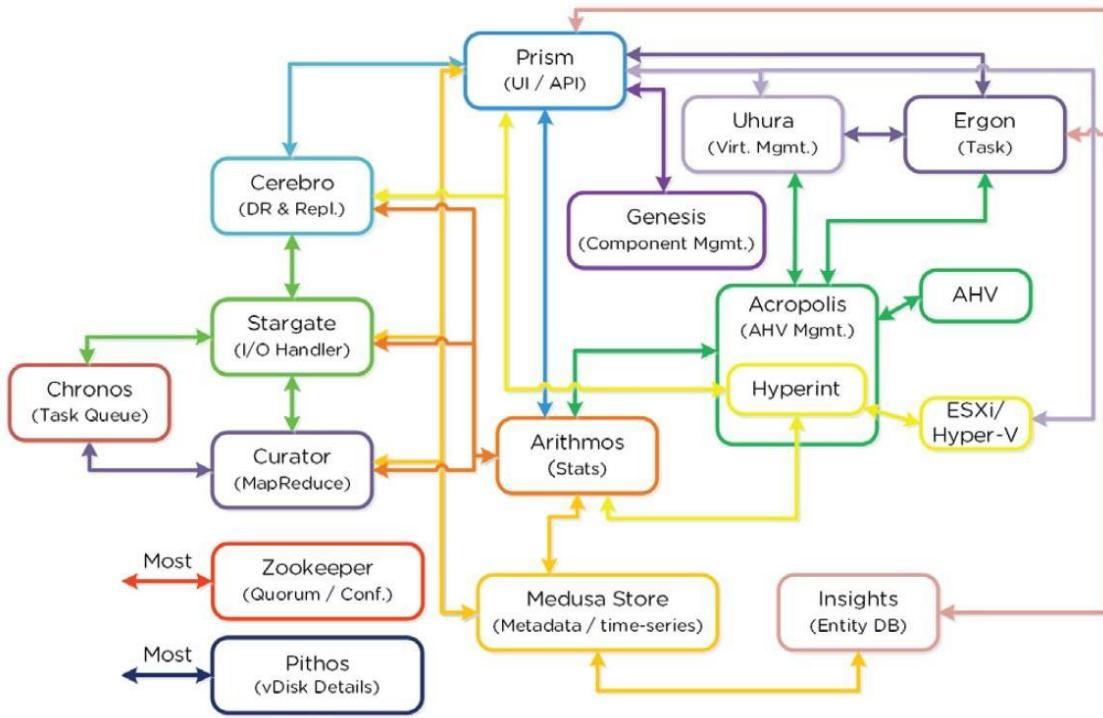


圖 . Nutanix 的集群組件

Cassandra

- 關鍵角色: 分散式中繼資料存儲
- 描述 : Cassandra 基於修改過的 Apache Cassandra，以分散式環的方式存放和管理所有的集群中繼資料。Paxos 演算法被用來保證嚴密的一致性。在集群中所有節點上都運行著這個服務。Cassandra 通過一個叫做 Medusa 的介面來訪問。

Zookeeper

- 關鍵角色: 集群配置管理
- 描述 : 基於 Apache Zookeeper 實現，Zookeeper 存放了所有的集群配置資訊，包括主機、IP 位址和狀態等。集群中有三個節點會運行此服務，其中的一個被選舉成 leader。Leader 接收所有請求並轉發到它的組員。一旦 leader 無響應，新的 leader 會被自動選舉出來。Zookeeper 通過稱作 Zeus 的介面來訪問。

Stargate

- 關鍵角色: 資料 I/O 管理
- 描述 : Stargate 負責所有的資料管理和 I/O 操作，是 hypervisor 主要的介面（通過 NFS、iSCSI 或 SMB）。該服務在集群中的每個節點上運行，以服務於當地語系化的I/O。



Curator

- 關鍵角色：以 Mapreduce 方式管理和清理集群
- 描述：**Curator** 負責管理和分佈整個集群中的任務，諸如磁片容量平衡、主動清理等。**Curator** 運行在所有節點上，受控於主 **Curator**（其負責任務委託）。**Curator** 有兩種掃描類型，每 6 小時一次全掃描和每 1 小時一次的部分掃描。

Prism

- 關鍵角色：使用者介面和 API
- 描述：**Prism** 是一個元件管理閘道，它讓管理員配置和監控 Nutanix 集群。它提供多種管理手段，如 Ncli、HTML5 UI 和 REST API。**Prism** 運行在集群中的每個節點，如同集群中其他元件一樣也採用 **leader** 選舉制。

Genesis

- 關鍵角色：集群元件和服務管理
- 描述：**Genesis** 是一個負責任何服務交互（啟動/停止等）以及初始配置的進程，運行在每個節點上。**Genesis** 是一個獨立於群集運行的進程，不需要配置/運行群集。它唯一的前提是 **Zookeeper** 必須正常啟動和運行著。**Cluster_init** 和 **cluster_status** 這兩個頁面所顯示的資訊由 **Genesis** 進程提供。

Chronos

- 關鍵角色：作業和任務調度
- 描述：**Chronos** 負責從節點掃描和節點間調度/節流任務中獲取作業和任務。**Chronos** 運行在每個節點上，受控於主 **Chronos**（負責任務委託且和主 **Curator** 運行在同一節點）。

Cerebro

- 關鍵角色：資料複製和容災管理
- 描述：**Cerebro** 負責 DSF 中的資料複製和容災管理部分，包含快照的調度、遠端網站的資料複製及網站的遷移和故障切換。**Cerebro** 運行在 Nutanix 集群的每個節點上，並且每個節點都參與遠端網站/集群的資料複製。

Pithos

- 關鍵角色：vDisk 配置管理
- 描述：**Pithos** 負責 vDisk (DSF 檔) 的配置資料。**Pithos** 構建於 **Cassandra** 之上，並運行在每個節點。

2.7 不間斷升級



對於 AOS 升級，需要執行以下幾個關鍵步驟：

2.7.1 步驟 1 - 升級前檢查

在升級前檢查期間，將驗證以下各項。注意：必須先成功完成此操作，然後才能繼續升級。

- 檢查 AOS，Hypervisor 版本之間的版本相容性
- 檢查集群健康狀況（集群狀態，可用空間和元件檢查（例如，Medusa，Stargate，Zookeeper 等）
- 檢查所有 CVM 和 Hypervisors 之間的網路連接

2.7.2 步驟 2- 將升級軟體上傳到 2 節點

完成升級前檢查後，系統會將升級軟體二進位檔案上載到群集中的兩個節點。這樣做是為了容錯，並確保一個 CVM 正在重新啟動，另一個 CVM 可供其他人從中提取軟體。

2.7.3 步驟 3 - 階段升級軟體

將軟體上載到兩個 CVM 後，所有 CVM 將並行進行升級。

CVM 具有兩個用於 AOS 版本的分區：

- 使用中的磁碟分割（當前正在運行的版本）
- 被動分區（進行升級的分區）

發生 AOS 升級時，我們將在非使用中的磁碟分割上執行升級。收到升級權杖後，它將升級後的分區標記為使用中的磁碟分割，並將 CVM 重新引導至升級後版本中。這類似於 bootbank /altbootbank。

注意：升級權杖在節點之間反覆運算傳遞。這樣可確保一次僅重啟一個 CVM。一旦 CVM 重新開機並穩定（檢查服務狀態和通信），權杖就可以傳遞到下一個 CVM，直到所有 CVM 都已升級。

升級錯誤處理

一個常見的問題是，如果升級不成功或在升級過程中部分出現問題，該怎麼辦？

如果發生某些升級問題，我們將暫停升級而不繼續進行。注意：這種情況很少發生，因為升級前檢查會在升級實際開始之前發現大多數問題。但是，如果升級前

檢查成功並且在實際升級過程中發生了某些問題，則不會對集群上運行的工作負載和使用者 I/O 產生影響。

Nutanix 軟體旨在在支援的升級版本之間以混合模式無限期地工作。例如，如果群集正在運行 **xyfoo** 並正在升級到 **xybar**，則系統可以在兩個版本的 CVM 上無限期運行。這實際上是在升級過程中發生的情況。

例如，如果您在 **xyfoo** 上有一個 4 節點群集，並開始升級到 **xybar**，則當第一個節點升級時，它將運行 **xybar**，而其他節點則在 **xyfoo** 上。此過程將繼續，並且 CVM 在收到升級權杖後將重新啟動到 **xybar** 中。

2.8 Foundation (鏡像)

2.8.1 架構

Foundation 是 Nutanix 提供的系統工具，用於引導，安裝鏡像和部署 Nutanix 集群。鏡像過程會安裝所需版本的 AOS 軟體和所選的 Hypervisor 軟體。

預設情況下，Nutanix 節點出廠會預裝 AHV 虛擬化平臺，如果需要不同的 Hypervisor 平臺，你必須用 **Foundation** 軟體重新對所有節點採用所需的 Hypervisor 進行重新鏡像操作。注意：某些 OEM 方式會從工廠直接安裝發運所需的 Hypervisor 軟體。

下圖展現的是 **Foundation** 的架構的高級視圖：

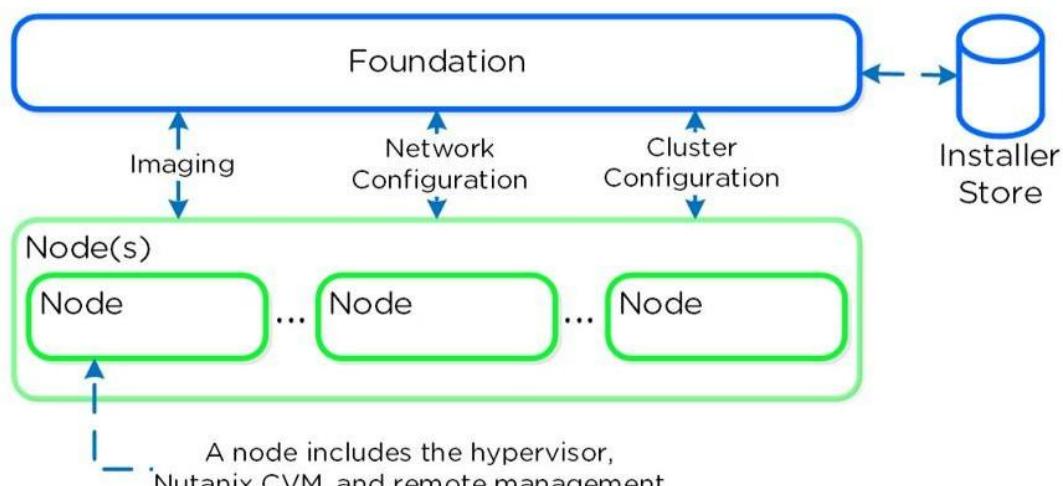
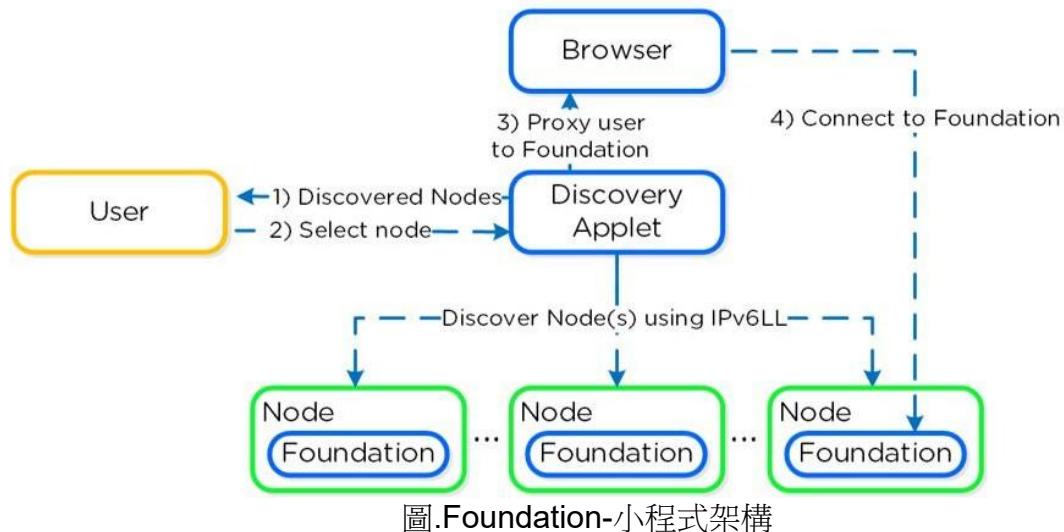


圖. Foundation-架構

從 4.5 版本起，為了簡化配置，CVM 裡面就內嵌了 **Foundation**。**Installer store** 是用來存放上傳的鏡像檔的目錄，這些鏡像可用于初始化部署時，也用於集群擴展時。

Foundation Applet 發現小程式(可以在此連結獲取)負責發現節點，並且允許用戶連接到某個選定的節點。一旦使用者選定了連接某個節點，小程式將本地主機埠：9442 代理到 CVM 的本地連接的 IPv6 位址埠：8000。

下圖展示了 Applet 架構的高級視圖



注意：發現小程序僅僅是發現和代理運行在所有節點上的 **Foundation** 服務的一種方式。所有鏡像和配置工作都由 **Foundation** 服務來完成，而不是小程序本身。

高級技巧：

如果你位於和目標的 Nutanix 節點（比如跨廣域網路）不同的網路（L2），而 CVM 已經配置了 IPv4 位址，你可以直接連接到 CVM 的 Foundation 服務（而不是用發現小程序）。

直接將流覽器連接到：`<CVM_IP>:8000/gui/index.html` 來訪問

2.8.2 輸入項

Foundation 工具配置時需要輸入如下資訊。典型的部署每個節點需要 3 個 IP 位址（hypervisor, CVM，遠端系統管理，如 IPMI，iDRAC 等）。除了每個節點的 IP 位址，建議設置一個集群和資料服務的 IP 位址。

- 集群
 - 名字
 - IP *
 - NTP *
 - DNS *



- CVM
 - CVM 的 IP
 - 遮罩
 - 閘道
 - 記憶體
 - Hypervisor
 - Hypervisor 主機 IP
 - 遮罩
 - 閘道
 - DNS*
 - 主機名稱首碼
 - IPMI *
- 節點 IP
- 遮罩
- 閘道

注意：'*'號項是可選項，但是強烈建議進行配置

2.8.3 系統鏡像和部署

第一步就是通過發現小程式（discovery applet）連到 Foundation 的使用者介面(如果是相同的二層網路，則無需節點 IP 位址)

Foundation Launcher				
Model	Serial Number	Position	Foundation	Status
NX-6020	13SM65350001	A	3.2.1	Free
NX-6035C	14SM62230008	A	3.2.1	Free
NX-6035C	14SM62230008	B	3.2.1	Free
NX-3000	QTFCE522600646	1	quanta	Free
NX-3000	QTFCE522600646	2	quanta	Free
NX-3000	QTFCE522600646	4	quanta	Free

圖.Foundatioon-發現小程式 (Discovery Applet)

如果無法發現所需的節點，請確認是否位於相同的二層網路中。

當連接到被選定的節點的 Foundation 實例，Foundation 主使用者介面就會顯示出來



Select All · Deselect All · Change RF (2)

Show only new nodes

Next >

圖. Foundation-發現頁面

下面展示的是所有發現的節點和他們所在的主機殼（Chassis）。選擇所需的節點組成集群，點擊“Next”

Select All · Deselect All · Change RF (2)

Show only new nodes

Next >

圖. Foundation-節點選擇

下一個頁面提示集群和網路輸入：

New Cluster Setup

Set up general information to create and connect your cluster to the network.

Cluster Information

Set up cluster level information like cluster name and IP address.

NAME TM3	NTP SERVER ADDRESS (OPTIONAL) 207.196.240.30
IP ADDRESS (OPTIONAL) 10.2.100.10	DNS SERVER IP (OPTIONAL) 10.1.1.100
<input checked="" type="checkbox"/> ENABLE IPMI	

Network Information

This is some basic information about your Hypervisor, CVM, IPMI IPs.

CVM NETMASK: 255.255.255.0	Hypervisor NETMASK: 255.255.255.0	IPMI (Optional) NETMASK: 255.255.255.0
----------------------------------	---	--

[◀ Prev](#) [Next ▶](#)

圖.Foudation-集群信息

New Cluster Setup

Set up general information to create and connect your cluster to the network.

NETWORK INFORMATION

This is some basic information about your Hypervisor, CVM, IPMI IPs.

CVM NETMASK: 255.255.255.0	Hypervisor NETMASK: 255.255.255.0	IPMI (Optional) NETMASK: 255.255.255.0
GATEWAY: 10.2.100.1	GATEWAY: 10.2.100.1	GATEWAY: 10.2.100.1
MEMORY: 32 GB	DNS SERVER IP: 10.1.1.100	

Post Imaging Tests

This enables a series of tests to ensure that the cluster has been correctly configured and everything is running smoothly.

ENABLE TESTING

[◀ Prev](#) [Next ▶](#)

圖.Foundation-網路資訊

當資訊輸入完成，點擊“Next”

下一步我們輸入節點的詳細資訊和 IP 位址：

Node Setup
Set up the IP addresses of your nodes.

Hostnames and IP Range
Specify the IP Range for the Nodes.

Hypervisor Hostname	CVM IP	Hypervisor IP
ENTER HOSTNAME TM3	FROM / TO 10.2.100.15 10.2.100.17	FROM / TO 10.2.100.11 10.2.100.13

IPMI IP (Optional)
FROM / TO
10.4.41.89
10.4.41.91

Manual Input
Manually fill in the IP Range for the Nodes.

[◀ Prev](#) [Validate Network ➔](#)

圖. Foundation-節點安裝
如果需要，可以手動的忽略主機名稱和 IP 位址

Node Setup
Set up the IP addresses of your nodes.

Manual Input
Manually fill in the IP Range for the Nodes.

QTFCE522600646

HYPERVERISOR HOSTNAME	CVM IP	HYPERVERISOR IP
1 TM3-1	10.2.100.15	10.2.100.11
2 TM3-2	10.2.100.16	10.2.100.12
3 TM3-3	10.2.100.17	10.2.100.13
4 TM3-4	10.2.100.18	10.2.100.14

IPMI IP (OPTIONAL)
10.4.41.89
10.4.41.90
10.4.41.92

[◀ Prev](#) [Validate Network ➔](#)

圖. Foudation-主機名稱和 IP
點擊“Validate Network”驗證網路配置並繼續進行.這一步檢查 IP 位址衝突確保連接正常



The screenshot shows the 'Node Setup' step of the Nutanix Foundation setup process. It displays fields for Hypervisor Hostname (TM3), CVM IP (10.2.100.05 to 10.2.100.17), and Hypervisor IP (10.2.100.11 to 10.2.100.13). An 'IPMI IP (Optional)' section is also present. A progress bar indicates 'Validating...' at 63%, with a note: 'Checking for IP conflicts and testing CVM-CVM and CVM-Host connectivity.' Buttons for 'Prev' and 'Next' are visible.

圖. Foundation-網路驗證

當網路驗證成功結束，我們現在就可以繼續選擇所需的鏡像。

為了把 CVM 上的 Acropolis 升級到一個更新的版本，需要從門戶上下載 Acropolis 並上傳打包檔。當我們有了 AOS 鏡像，下一步就是選擇 Hypervisor

對於 AHV 來講，它內置在 Acropolis 的鏡像中。對於其它的 Hypervisor 則需要上傳對應的鏡像檔。

注意：必須確保 AOS 和 Hypervisor 的版本在相容列表裡面（連接位址）。

當我們有了所需的鏡像之後，點擊“Create”：

The screenshot shows the 'Select Images' step. On the left, 'Acropolis' details are shown: Installed Version 4.7, uploaded tarball, and an 'Upload Tarball' button. In the center, 'Hypervisor' options are listed under AHV, ESX, HYPER-V, and CPS tabs. Available hypervisors include host-bundle-e16.nuta, host-bundle-e16.nuta, and kvm_host_bundle_2. On the right, 'SKU' selection is shown for Hyper-V, with 'Datacenter' selected. Buttons for 'Prev', 'Skip', and 'Create' are at the bottom.

圖. Foundation-選擇鏡像

如果不需要重新安裝鏡像你可以點擊'Skip'來跳過鏡像安裝過程。這樣就不會重新部署 Hypervisor 鏡像或者 Nutanix 集群的鏡像，僅僅對集群配置（如：IP 地址等）。

Foundation 將進行鏡像安裝（需要的情況下）和集群構建過程。



The screenshot shows the Nutanix Foundation interface for creating a cluster. At the top, there is a navigation bar with tabs: 1. Discover Nodes, 2. Define Cluster, 3. Setup Node, 4. Select Images, and 5. Create Cluster. The '5. Create Cluster' tab is currently selected. Below the navigation bar, a message says 'Showing Cluster creation progress'. There are two main sections: 'Overall Progress (1%) Cluster creation in progress.' and 'Cluster Creation Status'. The 'Cluster Creation Status' table has columns: STATUS, CLUSTER NAME, PROGRESS, and LOG. It shows one entry: TM3, Idle (0%), and a log link. Below this is a 'Node Status' table with columns: STATUS, HYPERVISOR IP, PROGRESS, and LOG. It shows one entry: 10.2.100.11, Running validations (1%), and a log link.

圖.Foundation-集群構建過程

當集群構建成功完成你將看到完成的介面：

The screenshot shows the Nutanix Foundation interface after a successful cluster creation. The navigation bar at the top is identical to the previous screenshot. The main area displays a message 'Cluster creation successful!' above a Nutanix logo icon. Below the logo, there is a link 'Manage your cluster with Prism.' and a 'Export logs' button.

圖. Foundation-集群構建完成

2.9 磁碟機解構

這一章節，我們將涉及到各種存放裝置（SSD/HDD）如何在 Nutanix 平臺被解構，分區和使用。注：所有容量單位是二進位位元組(GiB)而不是十進位位元組(GB)，同時把驅動器格式化的檔案系統和相關的消耗也一併考慮在內。

性能設備

性能設備是節點中性能最高的設備。這些可以是 NVMe 或 NVMe 和 SSD 設備的混合。它們存儲了一些關鍵專案，有更詳細的說明：

- Nutanix 主目錄 (CVM 核心)
- Cassandra (中繼資料 存儲)
- OpLog (持久寫緩衝)
- 內容緩存 (SSD 緩存)
- 擴展存儲 (持久存儲)

下圖示例了一個 Nutanix 節點的性能設備的存儲分析：

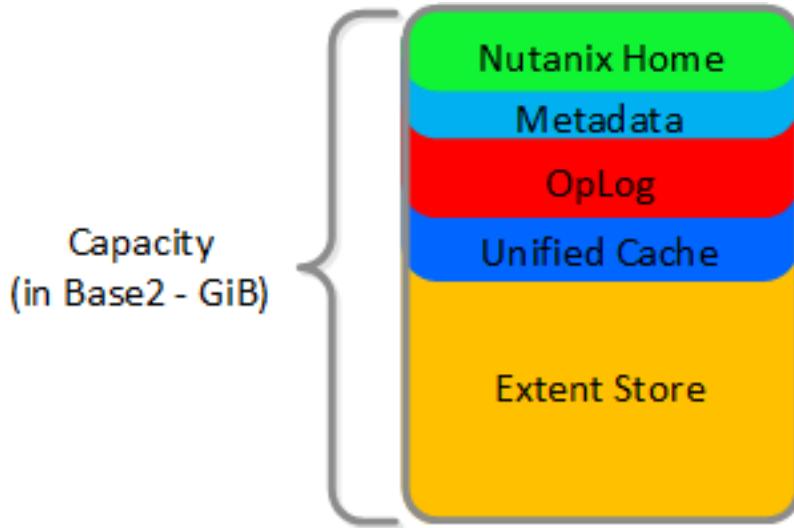


圖. SSD 驅動分解

上圖中，圖形和份額不是按比例繪製的。當評估剩餘 GiB 容量時，採用自頂而下的方法。例如，用於 OpLog 計算的剩餘容量(GiB) 應該是 SSD 被格式化後的容量減去 Nutanix Home 和 Cassandra 佔用的容量。

Nutanix Home 檔在前兩個 SSD 鏡像存儲以保證高可靠，並預留 60GiB 空間給中繼資料。

5.0 版本中 Cassandra 是分片在不同節點的 SSD 中的（目前最多 4 個），每個 SSD 預留 15GiB 的容量（如果中繼資料使用增加，可以利用部分 Stargate 的 SSD）。在雙SSD 系統，中繼資料會在 SSD 間鏡像。每個 SSD 預留 15GiB（雙 SSD 時 30GiB，4 個 SSD 時 60GiB）給中繼資料。

在 5.0 之前，Cassandra 默認在第一個 SSD，如果 SSD 故障 CVM 將重啟，Cassandra 存儲於第二個 SSD。此時，對於前兩個磁片，每個 SSD 的中繼資料預留是30GiB。



OpLog 分佈在所有 SSD 設備裡，每個節點最多 12 個 SSD (Gflag: max_ssds_for_oplog)。如果 NVMe 設備可用，OpLog 將被放置在這些設備上，而不是 SATA SSD。

可以使用以下公式計算每個磁片的 OpLog 預留： $\text{MIN}((\text{Max cluster RF}/2)*400 \text{ GiB})/\text{numDevForOplog}$ ， $((\text{Max cluster RF}/2)*25\%) \times \text{剩餘 GiB}$ 。注意：從版本 4.0.1 開始，OpLog 的大小調整是動態進行的，這將允許 extent store 存儲部分動態增長。使用的值假定完全使用了 OpLog。

例如，如果 RF2 (FT1) 集群中有 8 個 1TB 的 SSD 設備，結果將是：

- $\text{MIN}(((2/2)*400 \text{ GiB})/8), ((2/2)*25\%) \times \sim900\text{GiB} == \text{MIN}(50, 225) == 50 \text{ GiB}$ ，每個設備預留 50 GiB 紿 OpLog。

對於 RF3 (FT2) 集群，這將是：

- $\text{MIN}(((3/2)*400 \text{ GiB})/8), ((3/2)*25\%) \times \sim900\text{GiB} == \text{MIN}(75, 337) == 75 \text{ GiB}$ ，每個設備預留 75 GiB 紿 OpLog。

如果 RF2 (FT1) 集群有 4 個 NVMe 和 8 個 1TB 的 SSD 設備，結果會是：

- $\text{MIN}(((2/2)*400 \text{ GiB})/4), ((2/2)*25\%) \times \sim900\text{GiB} == \text{MIN}(100, 225) == 100 \text{ GiB}$ ，每個設備預留 100 GiB 紿 OpLog。

Extent Store 容量將是考慮所有其他預訂之後的剩餘容量。

HDD 設備

由於 HDD 設備主要作用於大型存放區，解構起來相對簡單：

- Curator 保留 (Curator 存儲)
- 擴展存儲 (持久存儲)

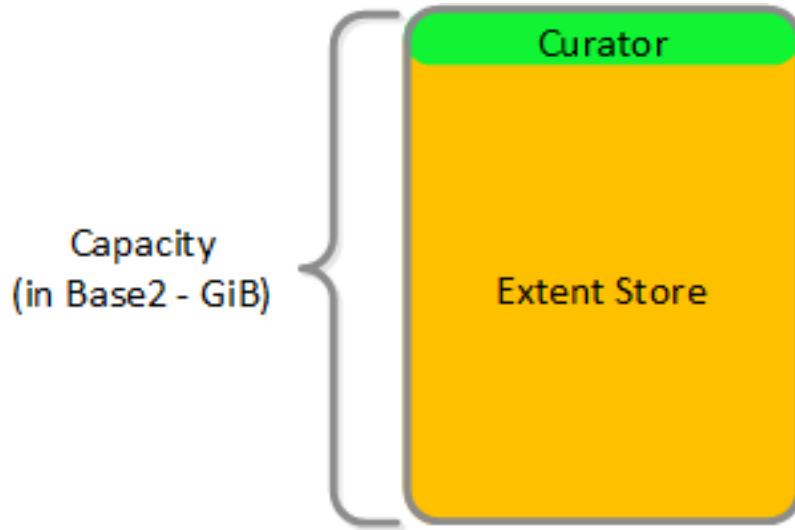


圖. HDD 驅動解析

3 第三部分：Prism

名詞解釋：**Prism**，控制台，為資料中心運營提供一鍵式管理和操作介面。

3.1 設計方法

建設一個美觀的、易於使用且直觀的產品是 Nutanix 平臺設計的核心思想，這一個章節將介紹我們的設計方法以及如何進行反覆運算。
 （省略部分確認）

3.2 架構

Prism 是一個分散式的資源管理平臺，允許用戶跨 Nutanix 集群環境管理和監控物件及服務，無論它們是在本地還是雲端。這些能力被分為兩種類別：

- 介面能力
 - ✓ HTML5 UI, REST API, CLI, PowerShell CMDlets 等
- 管理能力
 - ✓ 平臺管理，虛擬機器/容器的增刪查改，策略定義與合規，服務設計與狀態，分析和監控

下圖重點解釋了作為 Nutanix 平臺一部分的 Prism 概念特徵：

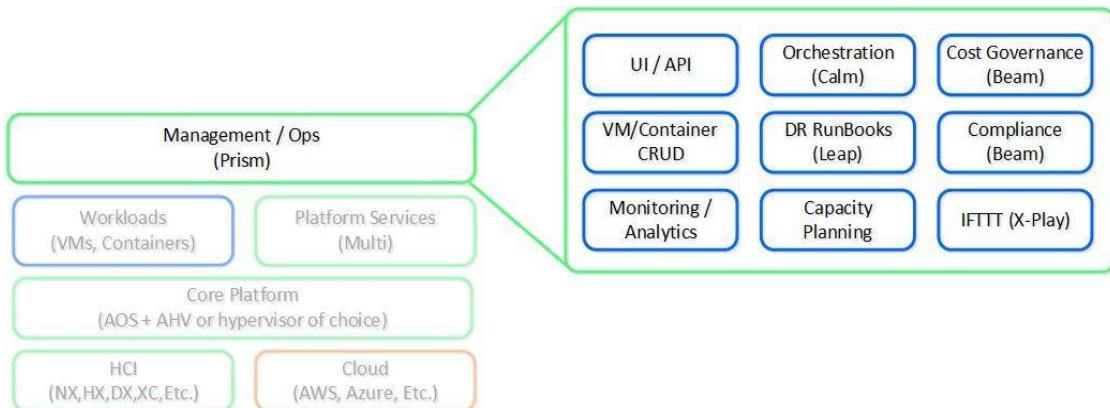


圖. Prism 的概要架構

Prism 主要由兩個組件構成：

- Prism Central (PC)
 - ✓ 多集群管理器，負責管理多個 Acropolis 集群，以提供單一的、集中的管理介面。Prism Central 是一個可選的軟體設備（虛擬機器），可以部署在 Acropolis 集群內或集群外部
 - ✓ 一對多的集群管理
- Prism Element (PE)
 - ✓ 本地集群管理者，負責本地集群的管理和運行，每個 Acropolis 集群都有自己內置的 Prism Element
 - ✓ 一對一的集群管理

下圖指出在 Prism Central 和 Prism Element 之間的邏輯關係：

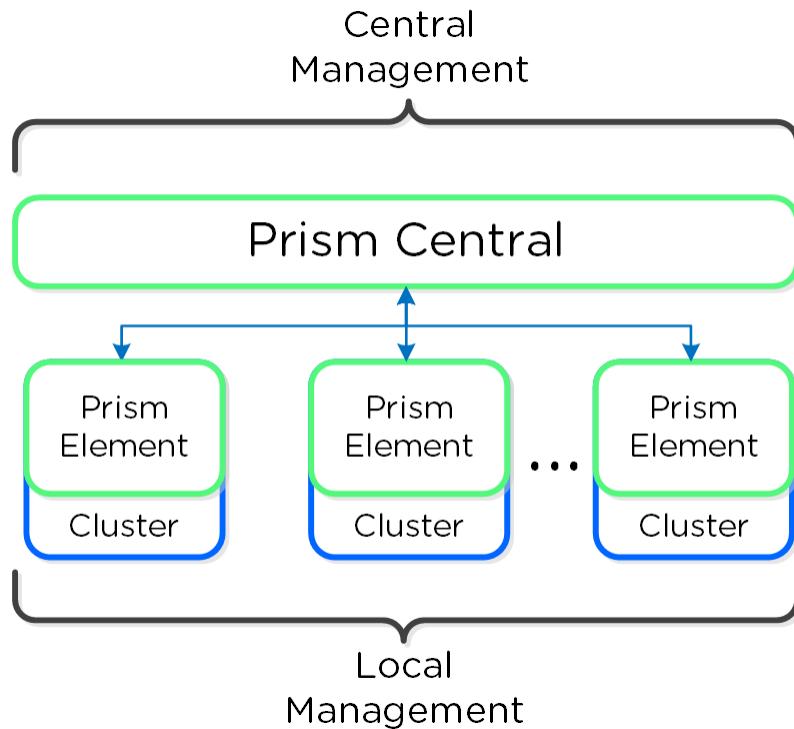


圖.Prism 架構

提示：

對於大規模或者分散式部署（例如多個集群或多個網站），建議使用 Prism Central 來簡化管理操作，為所有集群/網站提供單個管理介面。

3.2.1 Prism 的服務

Prism 服務運行在每一個 CVM 之上，其中一個 CVM 上的 Prism 服務會被選為 Prism Leader 元件，負責處理集群的 HTTP 請求。與其他的元件有 Master 類似，如果一個 Prism Leader 出問題，一個新的 Leader 會被選出。當不是 Prism Leader 的 CVM 得到 HTTP 的請求，它將永遠重定向請求到當前的 Prism Leader，並且返回 HTTP 應答碼 301。下圖是解釋 Prism 服務是如何處理 HTTP 請求：

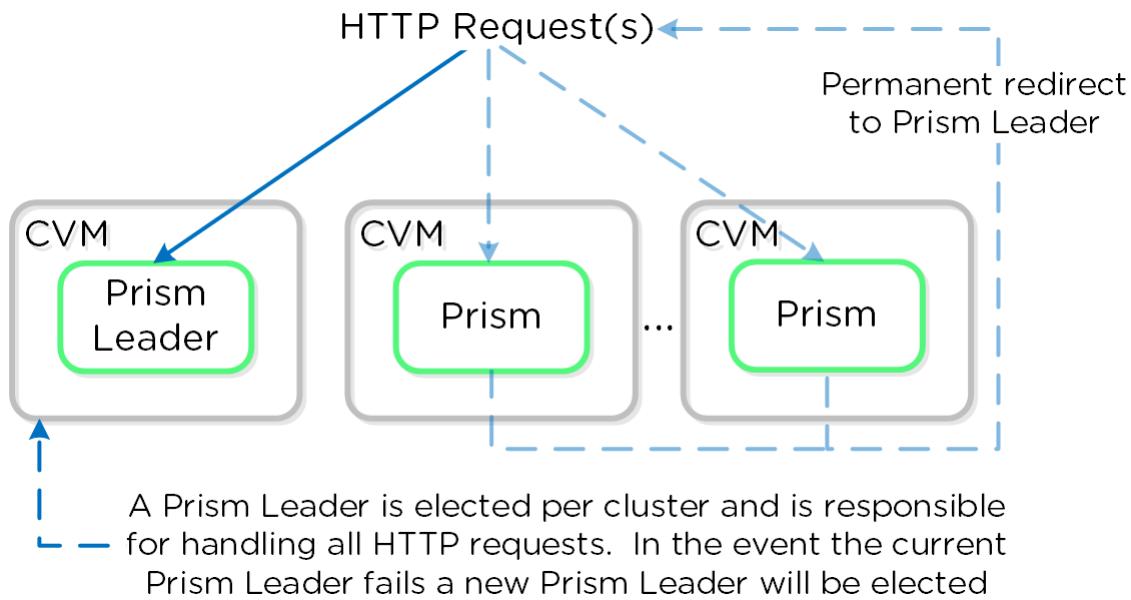


圖.Prism 服務處理 HTTP 請求

Prism 埠：Prism 監聽 80 和 9440 埠，如果 HTTP 流量來自於 80 埠，它將會被重定向到 9440 埠。

如果我們使用集群外部 IP（推薦），它將一直被當前的 **Prism Leader** 所有。即使 **Prism Leader** 出問題，集群 IP 也會轉移到新選舉出的 **Prism Leader** 上。一個 **gratuitous ARP** (gARP) 將被用來清理舊的 ARP 緩存條目。在上面的場景中，任何時候集群 IP 都可以被用來訪問 Prism，而無需重定向需要，因為 **Prism Leader** 已經被選出。

專家提示：可以在任一個 CVM 裡面使用命令將其設定為當前的 **Prism Leader**，請運行：“curl localhost:2019/prism/leader”

3.2.2 認證和存取控制

認證

Prism 當前支援如下認證方式：

- Prism Element (PE)
 - Local
 - Active Directory
 - LDAP
- Prism Central (PC)
 - Local
 - Active Directory
 - LDAP
- SAML Authn (IDP)

SAML / 2FA



SAML Authn 允許 Prism 集成哪些遵循 SAML 標準的外部認證提供者(IDP) (例如. Okta, ADFS, 等).

同時也可以允許你採用多因素認證 (MFA) 或雙因素認證 (2FA) 用於用戶登陸 Prism 的安全保證。

存取控制

敬請期待！

3.3 管理導航

Prism 提供相當直觀和簡單的使用方式，然而我們將介紹一些主要的頁面和基本用法。

Prism Central(如果部署)可以使用指定的 IP 位址或相應的 DNS 條目，而 Prism Element 可以通過 Prism Central 點擊相應集群條目導航到，或者通過任意 Nutanix CVM 或集群 IP (首選) 來訪問。一旦頁面被載入將登陸到 Prism 歡迎介面，可以使用 Prism 帳戶或 AD 憑證登陸。

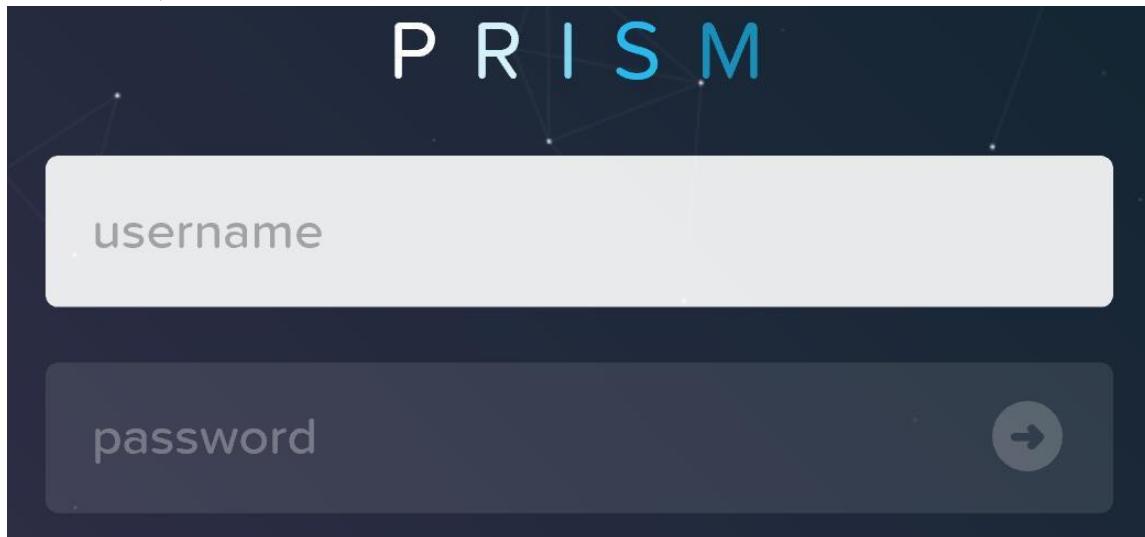


圖.Prism 登陸頁面

成功登錄後，您將被導航到控制介面頁面，該頁面將提供 Prism Central 中託管集群的概述資訊，或者是 PrismElement 中的本地集群。

在後面的章節中，我們會詳細介紹 Prism Central 和 Prism Element 的更多細節。

3.3.1 Prism Central

下圖顯示出 Prism Central 的儀錶盤。在這裡我們可以監控和管理多個集群：

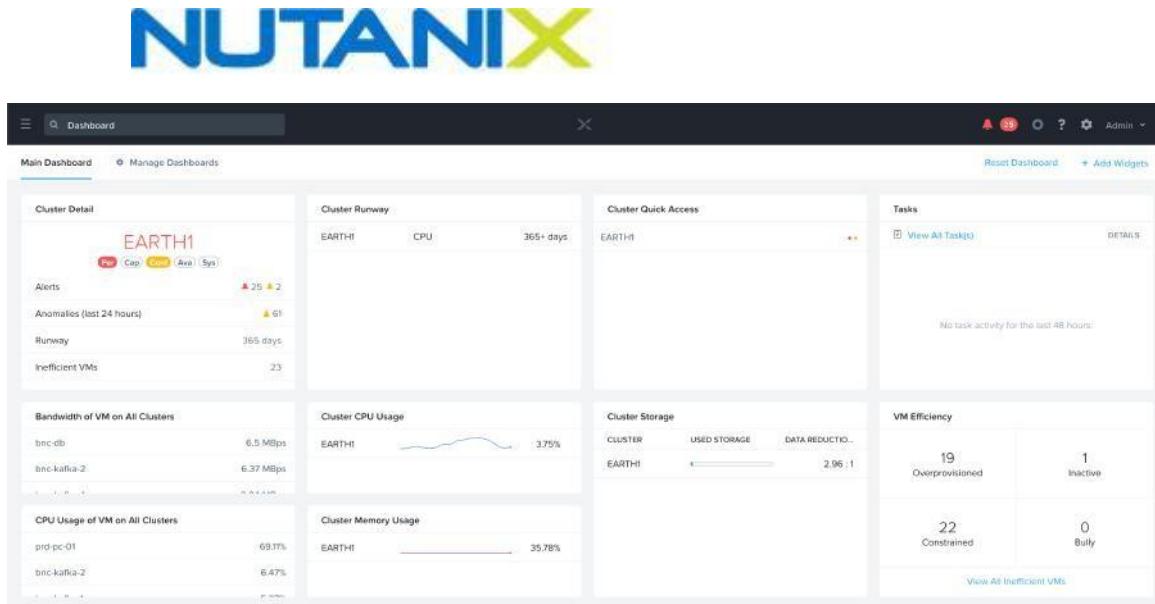


圖 Prism Central 儀錶盤

在儀錶盤上可以監控到所有的環境狀態，如果有任何告警或選項，可以深入查看。

Prism Central 包括如下主要頁面（注意：比較建議採用搜索的方式來定位到所需的元件）：

- 主頁面
 - 整體環境的監控儀錶盤，包括服務狀態、容量計畫、性能、任務等詳細資訊，為了獲得更多資訊，可以點擊每一個條目
- 虛擬架構
 - 虛擬化條目（例如：虛機，容器，鏡像，目錄，等等）
- 策略
 - 策略管理和創建（例如：安全性原則（FLOW），資料保護策略（備份/複製），恢復策略（容災），虛機工具策略（NGT）等）
- 硬體
 - 物理設備管理（例如：集群，主機，磁片，GPU）
- 活動
 - 整個環境的告警、事件和任務
- 操作
 - 操作面板、報告和動作（X-Play）
- 管理
 - 環境組織管理（例如：用戶，組，角色，可用域等）



- 服務
 - 附加服務管理（例如 Calm、Karbon）
- 設置
 - Prism Central 的配置

訪問這個功能表可以點擊這個三層的圖示：

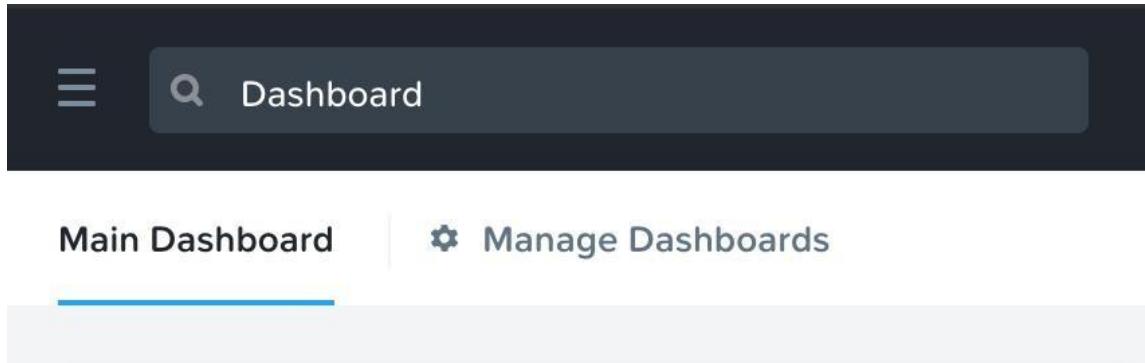


圖.Prism Central 功能表圖示

功能表會展開顯示為下面的可用選項。

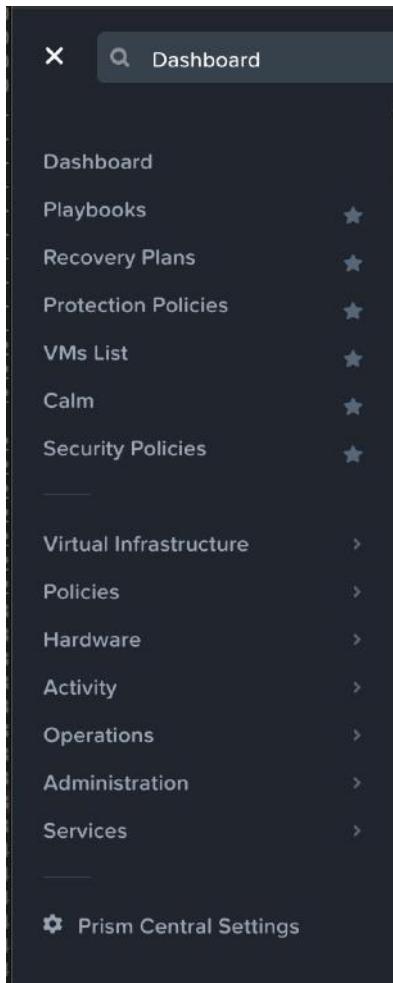


圖 Prism Central 菜單清單

3.3.2 搜索

搜索目前是一個 Prism Central 圖形介面的主要導航機制（功能表同樣有效）。可以在頂端左邊靠近功能表圖示的搜索欄中搜索導航到你想訪問的組件。

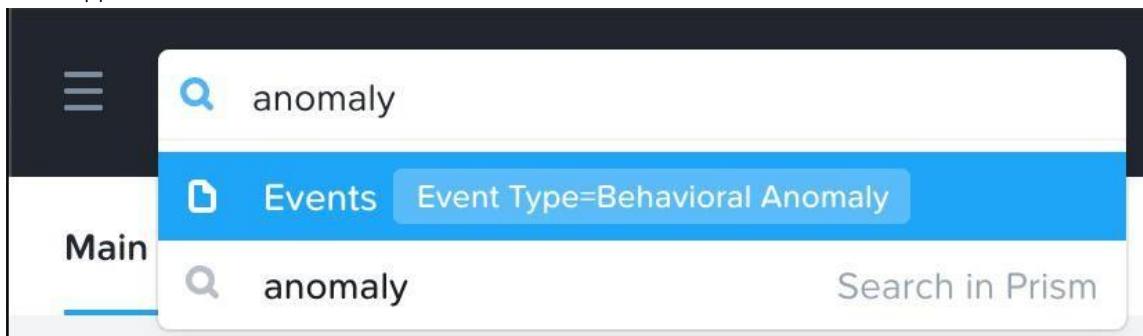


圖 Prism Central 搜索欄



搜索語法的示例

Prism Central 的搜索工具允許通過很多語法組合來豐富搜尋能力，示例如下：

Rule	Example
Entity type	vms
Entity type + metric perspective (io, cpu, memory)	vms io
Entity type + alerts	vm alerts
Entity type + alerts + alert filters	vm alerts severity=critical
Entity type + events	vm events
Entity type + events + event filters	vm events classification=anomaly
Entity type + filters (both metric and attribute)	vm "power state"=on
Entity type + filters + metric perspective (io, cpu, memory)	vm "power state"=on io
Entity type + filters + alerts	vm "power state"=on alerts
Entity type + filters + alerts + (alert filters)	vm "power state"=on alerts severity=critical
Entity type + filters + events	vm "power state"=on events
Entity type + filters + events + event filters	vm "power state"=on events classification=anomaly
Entity instance (name, ip address, disk serial etc)	vm1, 10.1.3.4, BHTXSPWRM
Entity instance + Metric perspective (io, cpu, memory)	vm1 io
Entity instance + alerts	vm1 alerts
Entity instance + alerts + alert filters	vm1 alerts severity=critical
Entity instance + events	vm1 events



Entity instance + events + event filters	vm1 events classification=anomaly
Entity instance + pages	vm1 nics, c1 capacity
Parent instance + entity type	c1 vms
Alert title search	Disk bad alerts
Page name search	Analysis, tasks

以上的示例只是一部分語法，最好的熟悉方式就是去嘗試下。

3.3.3 Prism Element

Prism Element 包括如下主要頁面：

- 主頁面
 - 本地集群監控儀錶盤，包括了告警、容量、性能、健康度、任務等詳細資訊，可以點擊具體條目獲得更多的資訊
- 健康
 - 環境、硬體及被管理物件的健康及狀態資訊，也包括 NCC 健康檢查資訊
- 虛擬機器
 - 實現全面的虛擬機器管理、監控及創建、運行、更新和刪除等功能
 - 虛擬機器監控（非 Acropolis Hypervisor）
- 存儲
 - 容器管理、監控及創建、運行、更新和刪除等功能
- 硬體
 - 伺服器、磁片和網路管理、監控和健康檢查，也涵蓋了集群的擴展。
- 資料保護
 - 容災、公有雲對接和同城 Metro Availability 的配置，管理保護域的物件、快照、複製和恢復
- 分析
 - 具有事件關聯性的集群及管理物件的詳細性能分析
- 告警
 - 本地集群和環境的告警

主頁面將提供詳細的告警、服務狀態、容量、性能及任務等資訊，可以點擊任意條目獲取更多的資訊。下圖展現了一個 Prism Element 管理下的本地集群儀錶盤：

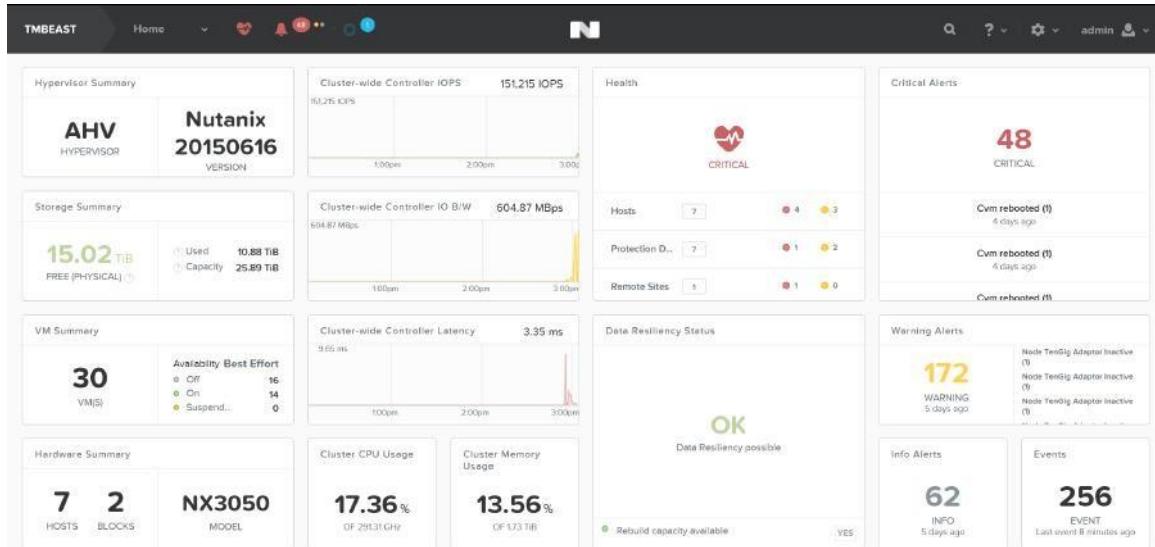


圖.Prism Element 儀錶盤

3.3.4 快速鍵

快速接入在 Prism 裡面至關重要，為了簡化用戶的使用允許使用者使用鍵盤的快速鍵來訪問不同的視圖和內容：

切換視圖：

- O – 概覽視圖
- D – 圖示視圖
- T – 表格視圖

活動和事件

- A – Alerts
- P – Tasks

下拉導航和菜單

- M – 功能表選項
- S – 設置選項
- F – 搜索條目
- U – 用戶條目
- H – 幫助

3.4 特性和使用



在下面內容中，我們將介紹一些典型的 Prism 應用程式以及一些常見的故障排除場景。

3.4.1 異常檢測

目前，在 IT 的日常操作中會有很多干擾因素。傳統的系統上會產生大量的告警，事件和通知。經常會導致操作者

- a) 迷失在大量的干擾資訊，而錯過嚴重的告警
- b) 丟棄告警和事件資訊

通過 Nutanix 的異常檢測能力，系統會監視各種基於時間序列的資料（例如：CPU 的使用，記憶體的使用，延遲等）並建立一個期望的值的範圍。只有命中範圍之外才會觸發告警和事件。你可以從任意的入口和事件頁看到異常事件。

後面的圖表展示的是許多 I/O 和磁片使用的異常，比較常見於在系統中執行大量的批次處理任務的情況。

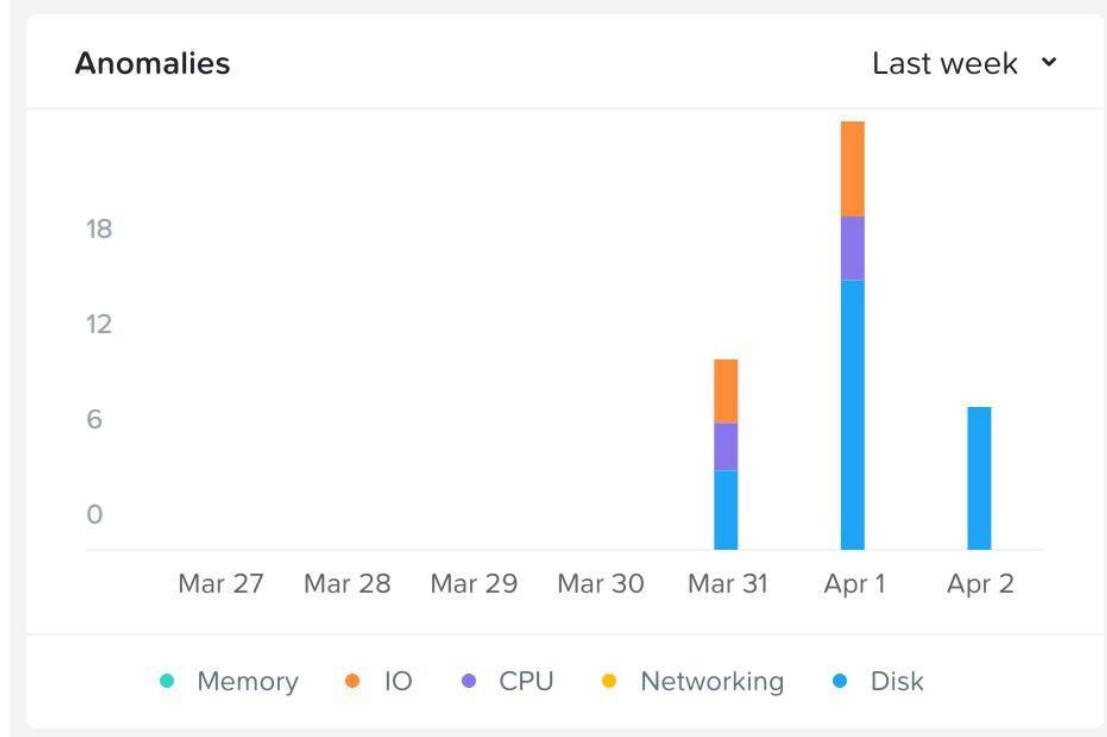


圖. Prism – 異常圖表

後圖展示了一個樣本度量的時間序列值的頻寬圖。



圖. Prism – 異常吞吐

這減少了不必要的警報，因為我們不希望系統經常在正常狀態告警。例如，由於緩存等原因，資料庫系統通常在 $>95\%$ 的記憶體利用率下運行。如果這個數字下降到10%，那就是異常，可能出了什麼問題(比如資料庫服務宕機)。

另一個例子在週末運行的批次處理的工作負載。例如，平常 I/O 頻寬可能在工作周期間很低，但是在週末一些批次處理進程(例如備份、報告等)運行時，I/O 可能會出現很大的峰值。我們的系統將檢測到這種週期性，並在週末提高頻寬標準。在這裡你可以看到一個異常事件發生了，因為值超出了預期的範圍：

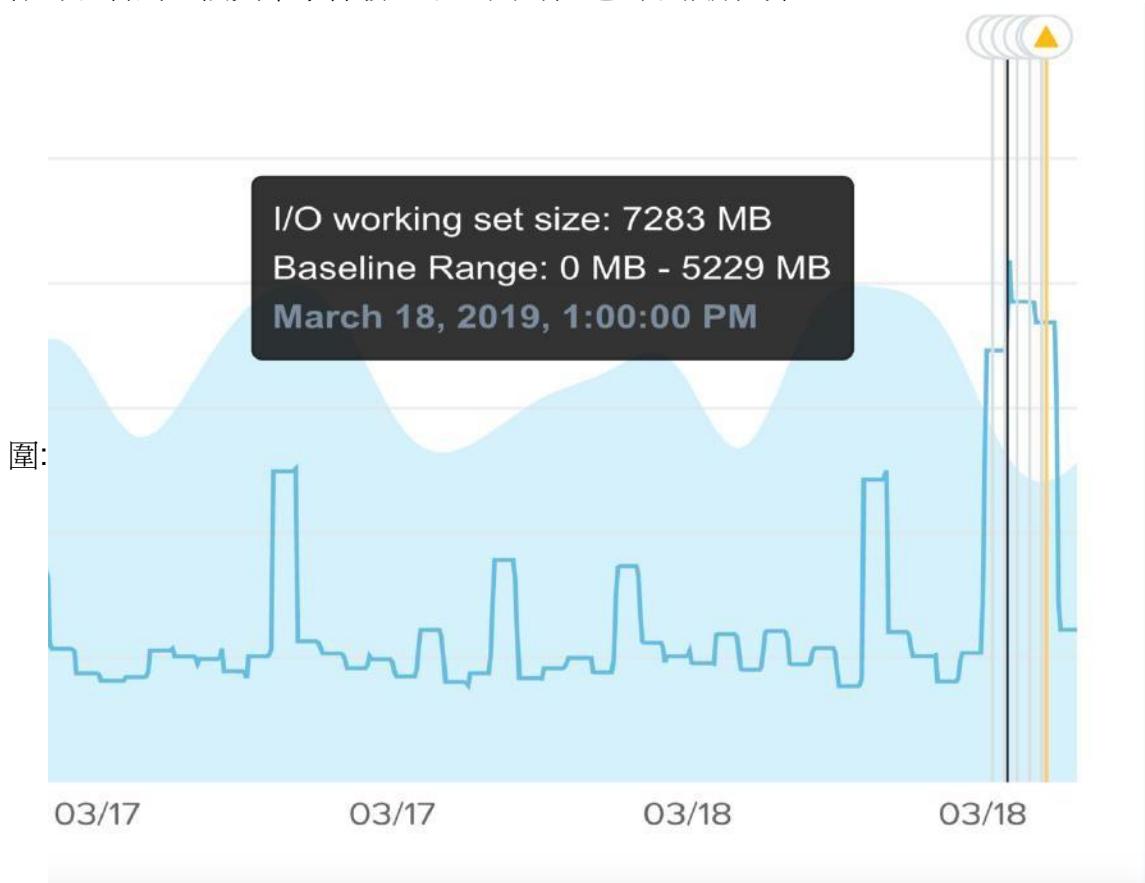


圖. Prism – 異常事件



另一個有趣的異常的情況是季節性。例如，在假日期間，零售商的需求將高於一年中的其他時間，或在月末關閉期間。

異常檢測解釋了這種季節性，並利用以下時間段來比較微觀(每日)和宏觀(季度)趨勢：

- 每日
- 每週
- 每月

您還可以設置自己的自訂警報或靜態閾值：

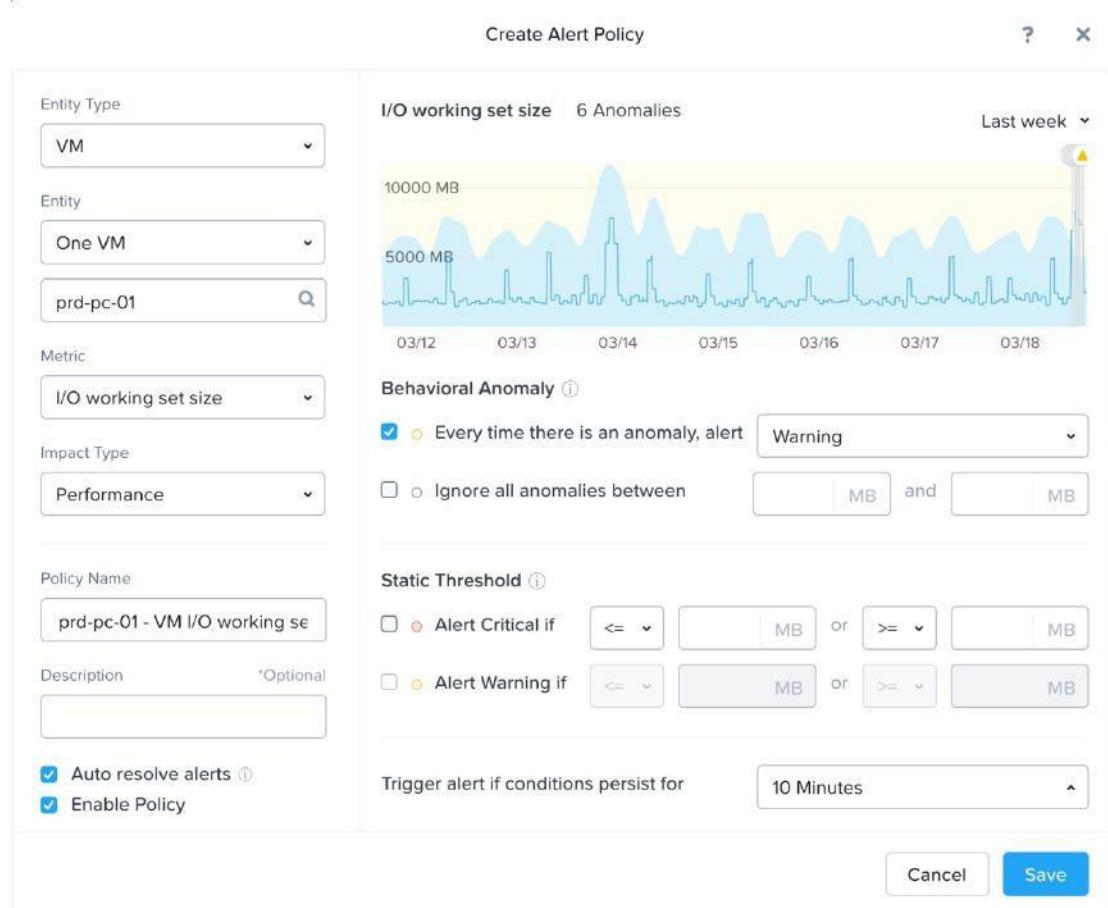


圖. 異常自訂事件

異常檢測演算法

Nutanix 利用一種確定頻段的方法稱為“通用極端故意偏差測試”。簡單理解類似於置信區間，該區間的值介於演算法確定的上下限之間。

該演算法需要 3 種顆粒度（例如每天，每週，每月等）來計算季節性和預期範圍。例如，為了適應每個季節，將需要以下資料量：

- 每天：3 天
- 每週：3 周（21 天）
- 每月：3 個月（90 天）

3.4.2 Nutanix 軟體升級

執行 Nutanix 軟體升級是一個非常簡單且無需中斷業務的過程。

在登錄到 Prism 以後點擊介面右上角的齒輪圖示，或者快速鍵 S 選擇“升級軟體”：

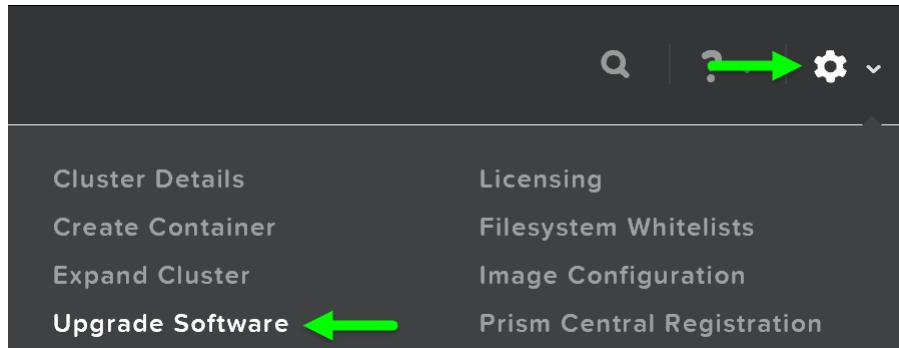
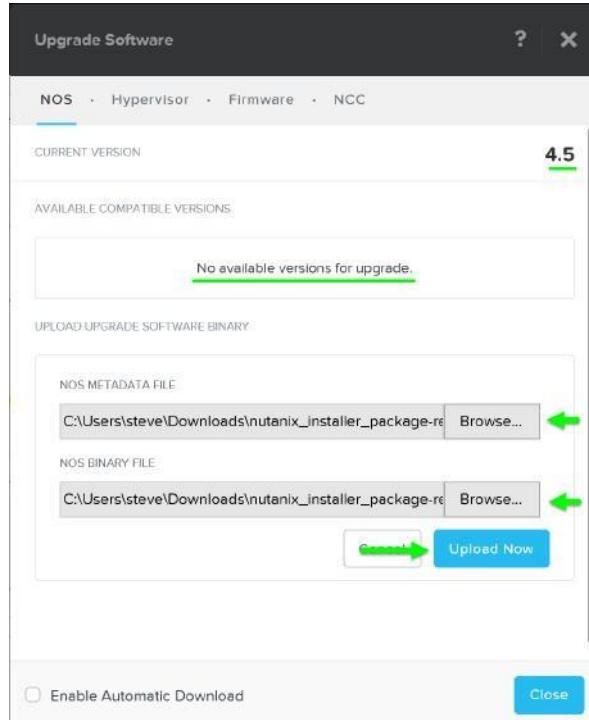


圖.設置更新軟

啟動“更新軟體”對話方塊，然後顯示出當前的軟體版本，是否有新的版本更新，也可以手動上載 NOS 檔更新軟體版本。如下圖所示，可以從外部雲環境下載更新版本，也可以手動上傳新版本軟體：





圖：更新軟體主介面

從 CVM 更新軟體

在一些特定場景下，你可能想手動下載軟體並上載到 CVM 裡。我一般在自己的環境中一般採用這種模式。
首先通過 SSH 的方式連結到任意 CVM 並找到 Prism Leader。

```
curl localhost:2019/prism/leader && echo  
再通過 SSH 連接到 Prism leader 下載套裝軟體和中繼資料的 JSON 檔。
```

運行下面的命令 "載入" 軟體到 Prism:
ncli software upload file-path=<PATH_TO_SOFTWARE> meta-file-path=<PATH_TO_METADATA_JSON>
software-type=<SOFTWARE_TYPE>
下面是一個完整的示例：



如果點擊上傳，則將更新軟體上傳到 Nutanix CVM 中：

AVAILABLE COMPATIBLE VERSIONS

4.5

uploading...

UPLOAD UPGRADE SOFTWARE BINARY

Uploading file...

圖.更新軟體上傳

在軟體上傳完成後點擊"Upgrade"按鈕，啟動更新流程：

AVAILABLE COMPATIBLE VERSIONS

4.5



Upgrade

圖.升級軟體——啟動升級

再次確認是否要更新軟體：



Do you want to upgrade to 4.5?



圖. 升級軟體——更新確認

更新過程啟動後進行更新檢查，然後自動更新軟體：

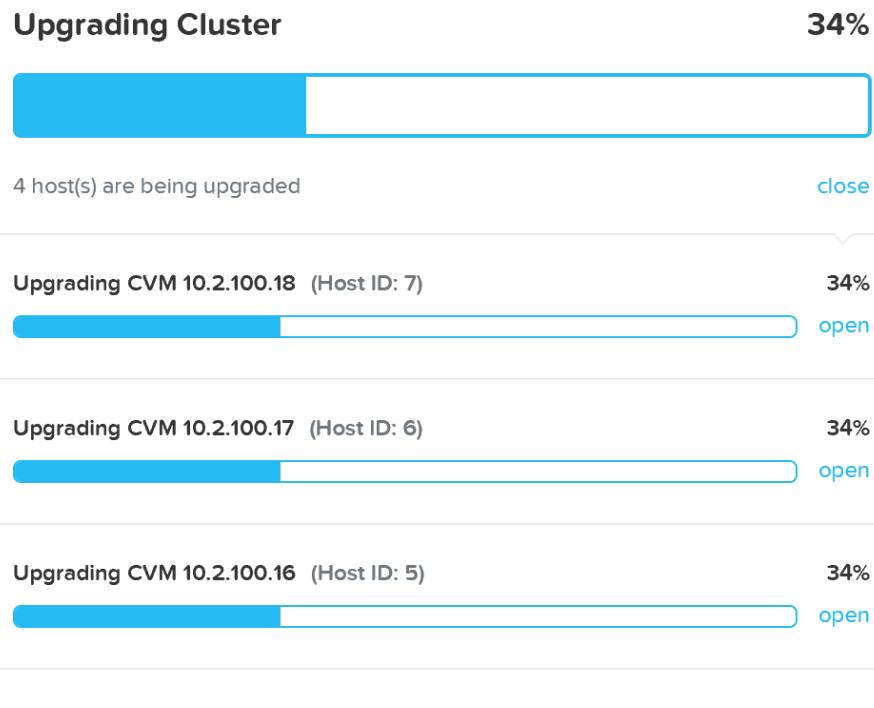


圖. 升級軟體-執行中

一旦更新過程完成，將看到更新狀態，隨後即刻可以訪問新的功能：

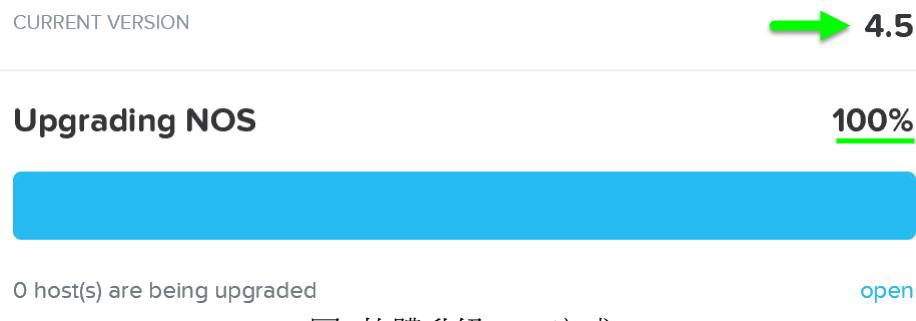


圖. 軟體升級——完成

提示：

當 Prism Leader 升級時，Prism 會話會出現暫時中斷，但所有的虛擬機器和服務運行不受影響。

3.4.3 Hypervisor 升級

類似 Nutanix 軟體升級，Hypervisor 的升級也能通過 Prism 採用完全自動化方式，其升級步驟與之前類似，運行彈出“Upgrade Software”對話方塊，然後選擇“Hypervisor”。Hypervisor 升級可以選擇從雲端自動下載軟體，也可以手動上傳新的軟體版本。

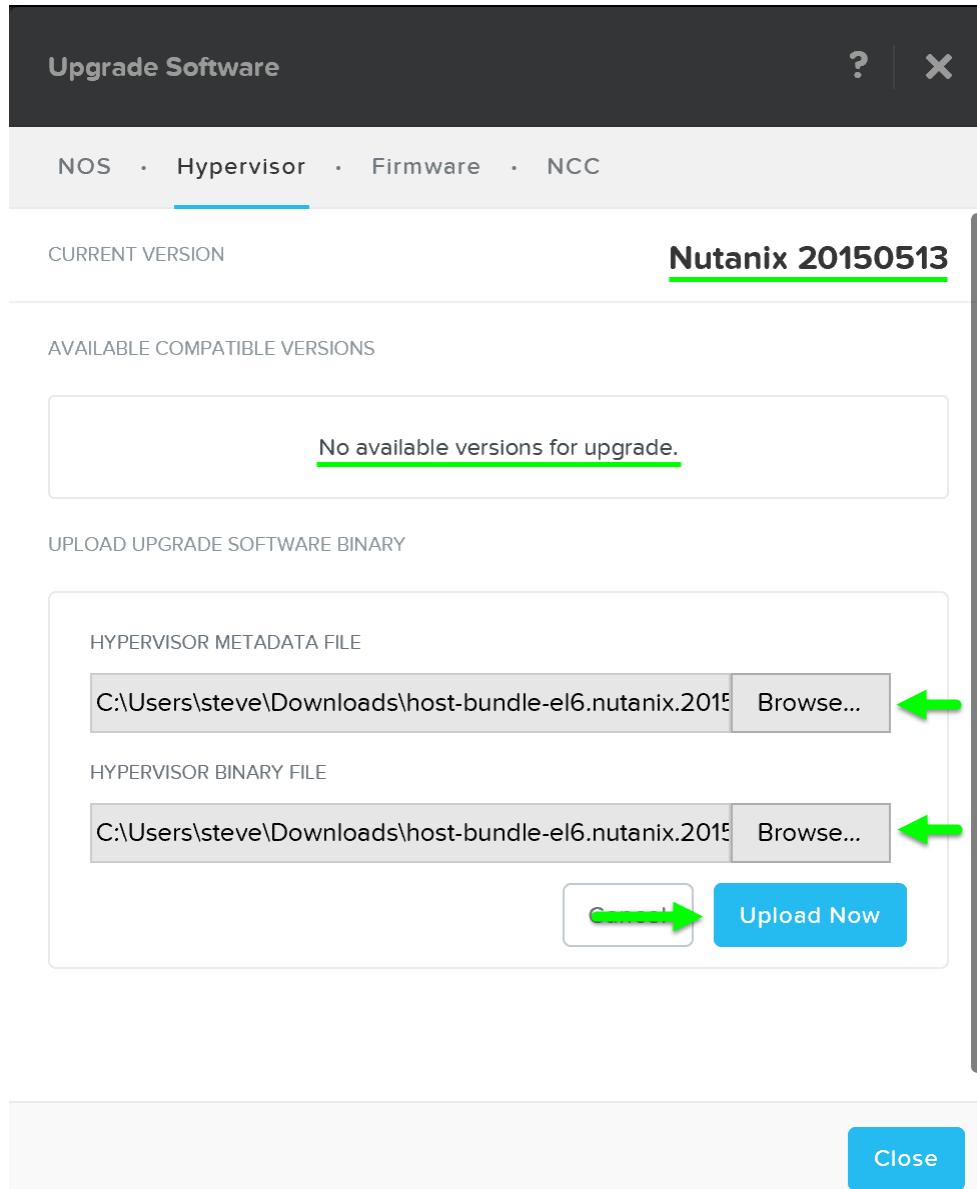


圖. 升級 Hypervisor — 主菜單

它將載入升級軟體到虛擬化層，一旦新版本軟體被成功載入，點擊“Upgrade”按鈕開始升級過程。



圖. 升級 Hypervisor — 開始升級過程

選擇再次確認 Hypervisor 升級，點擊“Upgrade”。

Do you want to upgrade to el6.nutanix.20150616?



圖. 升級 Hypervisor — 確認升級

系統將自動進行升級前的檢查，然後上傳 Hypervisor 進行整個集群的升級工作。

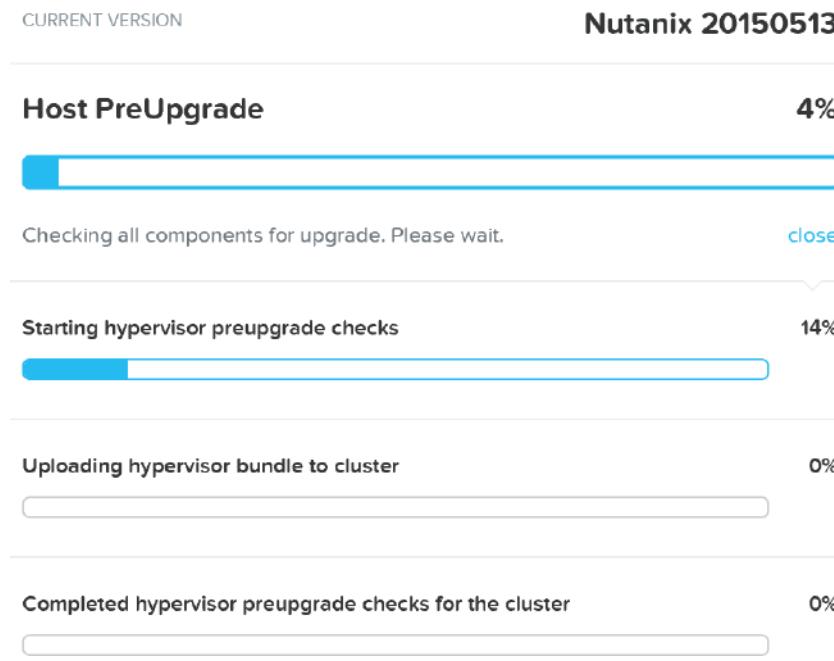


圖. 升級 Hypervisor — 預檢查過程

一旦升級前檢查完成，虛擬化層升級將以進度條形式開始進行：



CURRENT VERSION

Nutanix 20150513

Upgrading Hypervisor

22%



7 host(s) are being upgraded

[close](#)**Host Upgrade (Host ID: 9)**

14%

[open](#)**Host Upgrade (Host ID: 8)**

14%

[open](#)**Host Upgrade (Host ID: 7)**

71%

[open](#)**Host Upgrade (Host ID: 14)**

14%

圖. 升級 Hypervisor — 升級執行

類似於 Nutanix 軟體輪流升級的特性，集群的每個主機都會依次序在運行虛擬機器不受影響的情況下完成升級工作。當主機 Hypervisor 升級的時候，上面虛擬機器會被自動線上遷移到其他節點上；在 Hypervisor 升級過程中，主機會被自動重啟。這個過程會依次序進行，直到集群內所有節點都被升級完成。

提示：

可以通過 Nutanix CVM 運行命令“host_upgrade --status”查看集群升級的狀態，也可以通過查看 CVM 的日誌 “/data/logs/host_upgrade.out”獲取集群狀態資訊。

一旦升級完成，可以看見更新後的狀態，立刻可以訪問新的功能。



CURRENT VERSION

→ Nutanix 20150616

Upgrading Hypervisor

100%

0 host(s) are being upgraded

[open](#)

圖. 升級 Hypervisor — 升級完成

3.4.4 集群擴展(增加節點)

Acropolis 集群動態擴展能力是核心的功能。要擴展 Acropolis 集群，可以將節點上架、擺放和插線，然後啟動節點。一旦節點被啟動後，其會以 mDNS 方式自動被當前集群發現。

下圖展現了一個 7 節點的集群，動態發現 1 個新節點的加入。

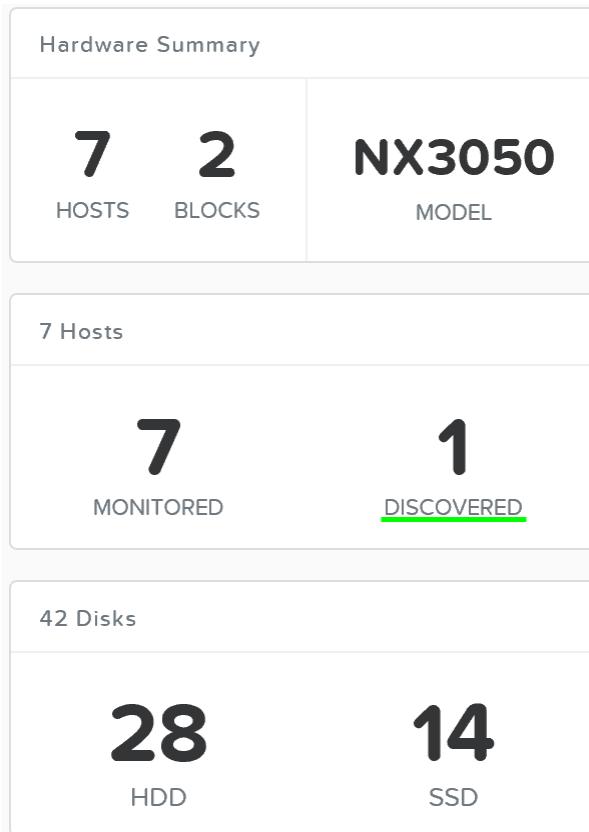


圖. 動態發現節點

多個節點可以被發現並被同時加入到集群中。一旦節點被發現，可以點擊主頁面右上角的“Expand Cluster”開始延伸這個集群。

Hardware

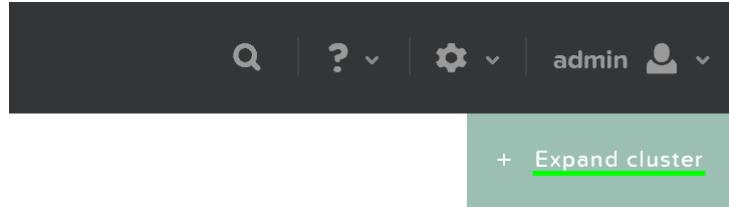


圖. Hardware 頁面中的 Expand Cluster

也可以在任意頁面通過點擊頁面頂端右上角的齒輪圖示，並找到 Expand Cluster。

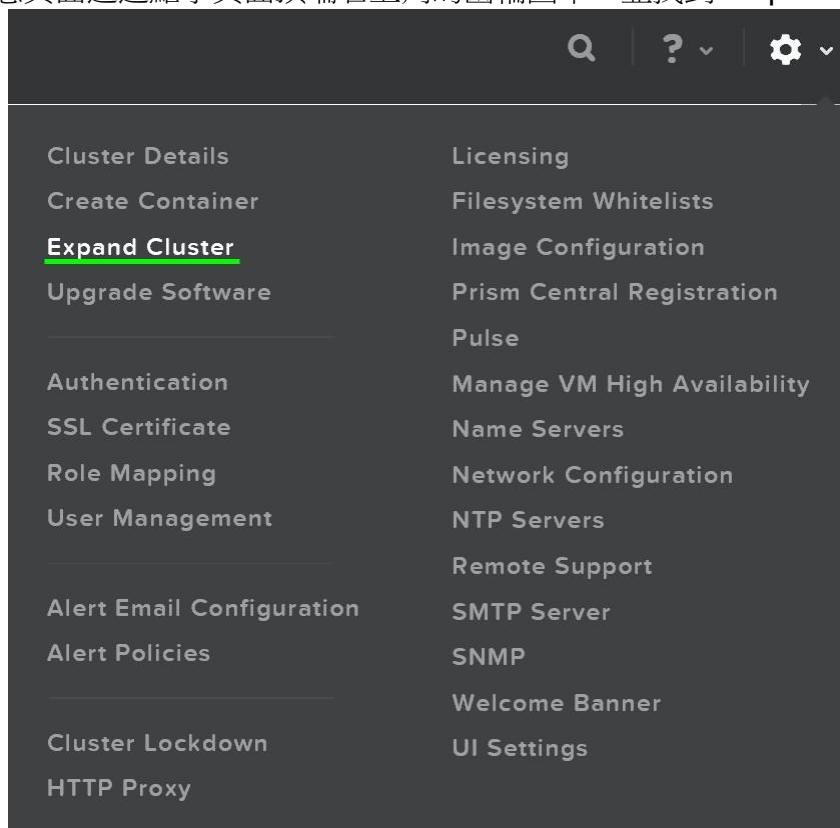


圖. 齒輪圖示下的 Expand Cluster

點開 Expand Cluster 功能表，可以選擇要增到集群內的節點和指定相應的 IP 位址。

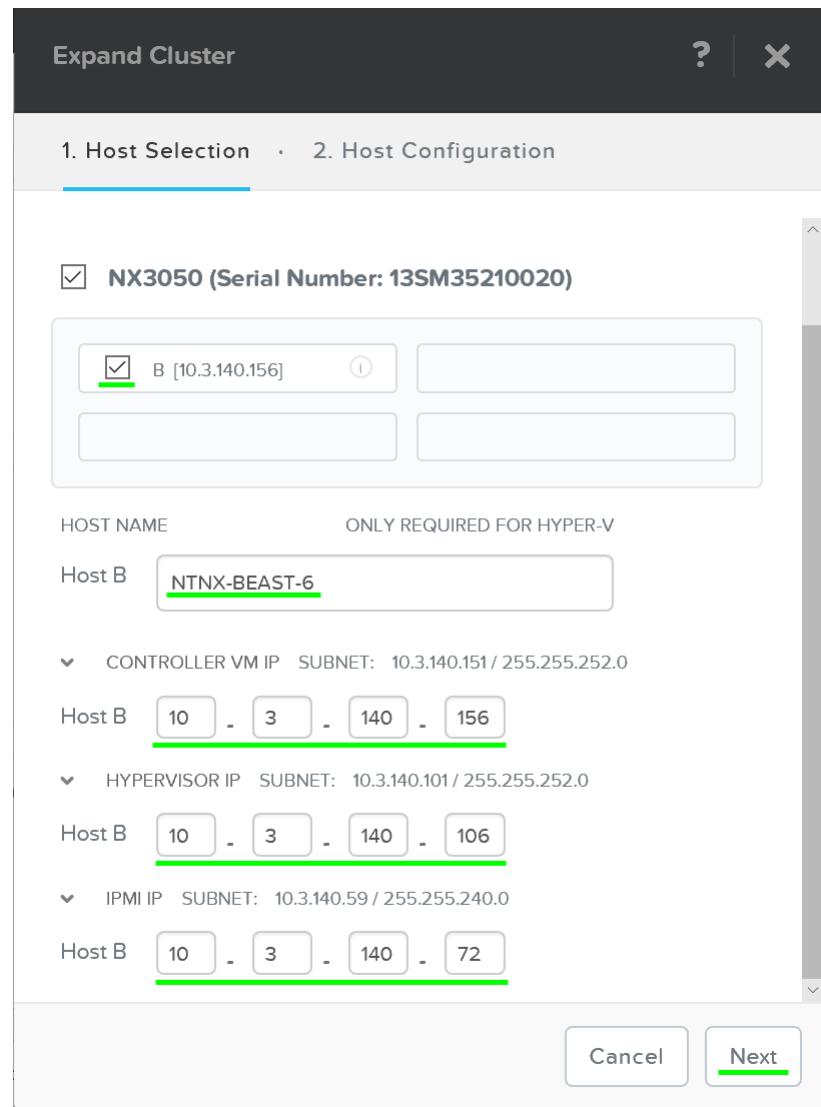


圖. Expand Cluster—主機選擇

在主機被選中後，將會被提示上傳 Hypervisor 鏡像，用來安裝新添加的節點。對於 AHV 或者鏡像已經存在於 Foundation 安裝資料夾，則沒有上傳的必要。



Expand Cluster ? | X

1. Host Selection · 2. Host Configuration

HYPERVERISOR(S) NEEDED

1 hypervisor is detected. Please upload correct ISO images respectively before expanding with selected hosts.

Hypervisor : AHV REQUIRED BY 1 HOST(S)

Uploading: host-bundle... X

Hypervisor ISO Whitelist UPDATED: JUST NOW

✓ whitelist.json [Update](#)

Back Cancel Validate **Expand Cluster**

圖. Expand Cluster—主機配置

在鏡像上傳完成後，可以點擊 **Expand Cluster** 啟動鏡像安裝和集群擴展任務。

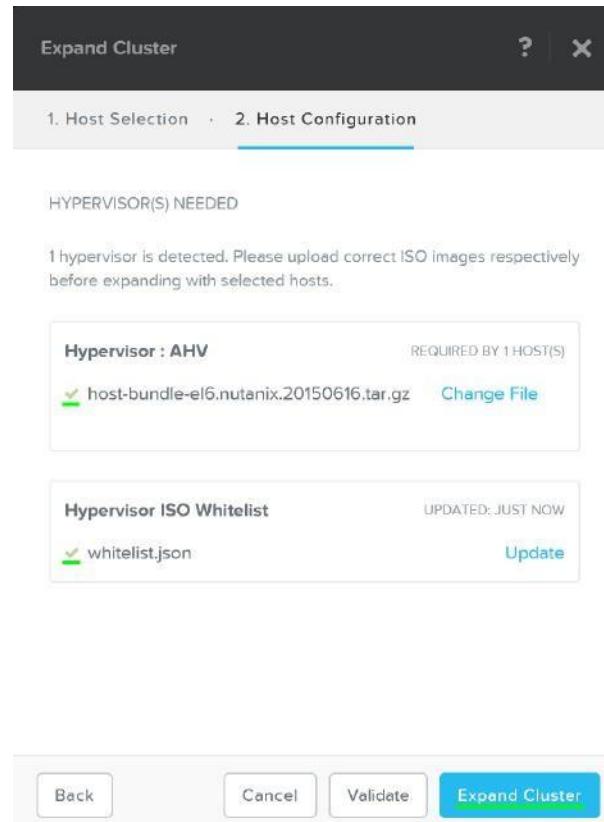


圖. Expand Cluster—執行

執行任務將被提交，在工作列將顯示任務執行的情況。

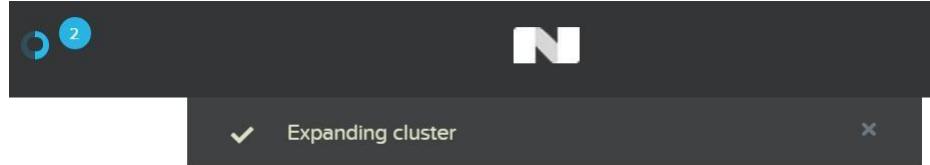


圖. Expand Cluster—執行

任務執行狀態可以在擴展任務狀態列裡查看。

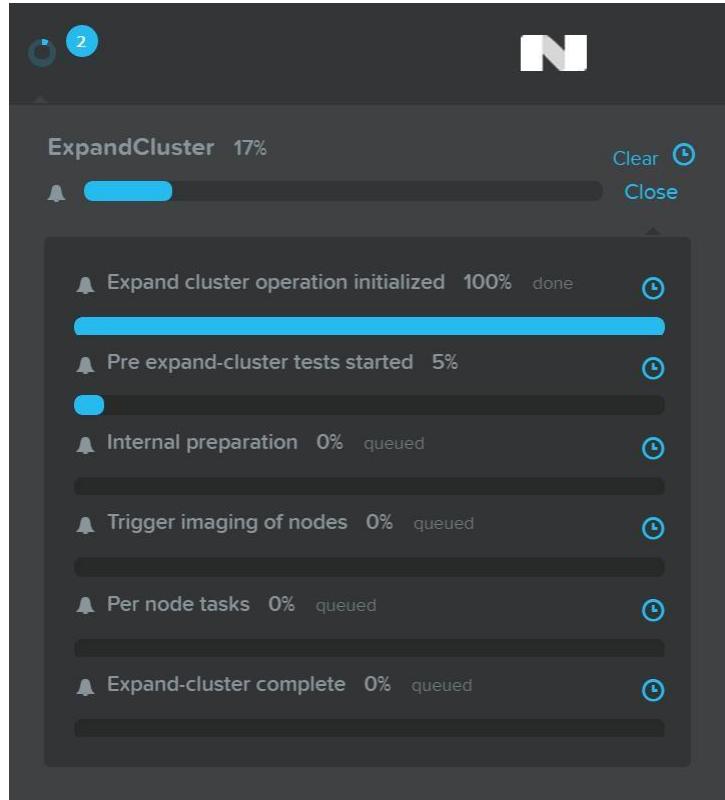


圖. Expand Cluster—執行

在鏡像安裝和添加節點的過程完成後，可以看到更新後的集群內節點數量和資源情況。

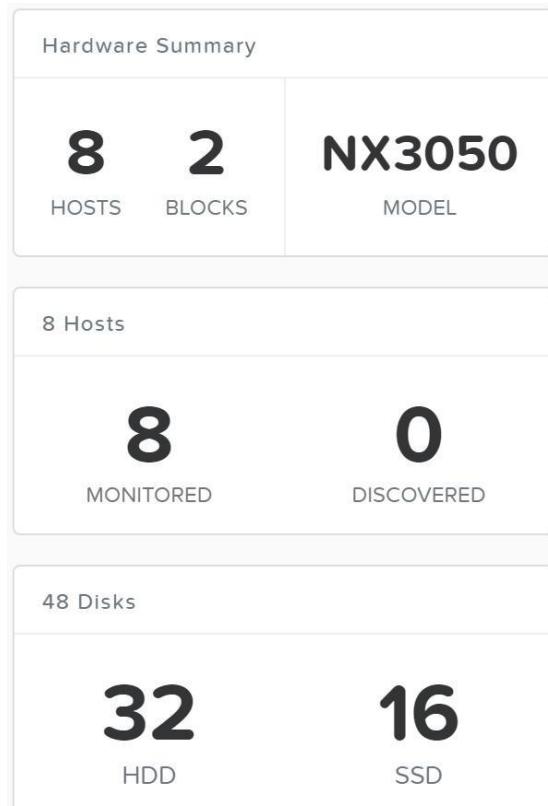


圖. Expand Cluster—執行

3.4.5 I/O 度量

識別瓶頸是性能故障診斷過程中的一個關鍵部分。

為了輔助這一過程，Nutanix 已經在“虛擬機器”頁面引入了一個新的 I/O 指標部分。延遲取決於多個變數。（佇列深度，I/O 規模，系統條件，網路速度等）

本頁面旨在提供對 I/O 規模、延遲、資源和模式的分析。

要使用這部分，請轉到“虛擬機器”頁面，並從表中選擇所需的虛擬機器。在這裡我們可以看到宏觀的使用度量：

VM NAME	HOST	IP ADDRESSES	CORES	MEMORY CAPACITY	STORAGE	CPU USAGE	CONTROLLER READ IOPS	CONTROLLER WRITE IOPS	CONTROLLER IO BANDWIDTH	CONTROLLER AVG IO LATENCY	BACKUP	FLASH MODE
loadgen1	NTNX-BEAST-5	10.314...	8	8 GiB	136.92 GiB / 4.14 TiB	28.45%	24799	10603	283.25 MBps	1.33 ms	Yes	No

圖. 虛擬機器頁面-細節

在這部分表格下面可以找到“I/O 度量”選項卡

Manage Guest Tools	Launch Console	Power Off Actions	Take Snapshot	Migrate	Pause	Clone	Update	Delete
VM Performance	Virtual Disks	VM NICs	VM Snapshots	VM Tasks	I/O Metrics	Console		

圖：虛擬機器頁面——I/O度量項

選擇“I/O 度量”選項卡後，將顯示詳細視圖。我們將詳細分析這一頁，以及如何在這一節中使用它。

第一個視圖是“平均 I/O 延遲”部分，它顯示了過去三小時的平均讀/寫延遲。

預設情況下，最新報告的值將及時與下面對應的詳細度量值一起顯示出來。

也可以使用滑鼠在表上移動來查看歷史延遲值，並點擊擊圖表的某一特定時間，在下方顯示詳細度量值



圖. I/O度量-延遲標圖

這在出現急劇的峰值負載時是很有說明的。如果你看到一個尖峰並想進一步調查，點擊尖峰並評估下面的細節

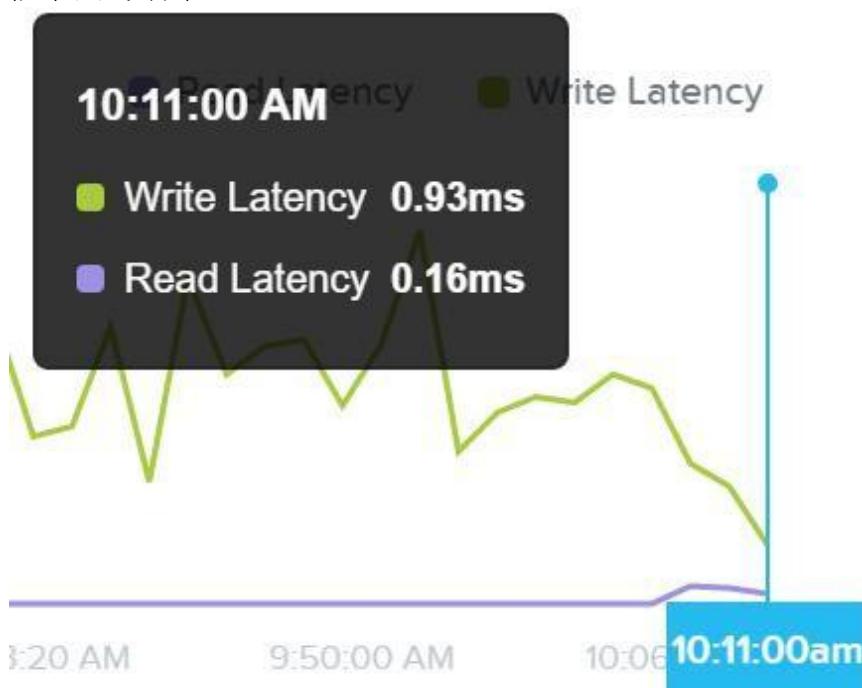


圖. I/O度量-延遲標圖

如果延遲都在接受範圍內，就不需要進一步挖掘了。

下一部分顯示為讀/寫 I/O 規模的長條圖：

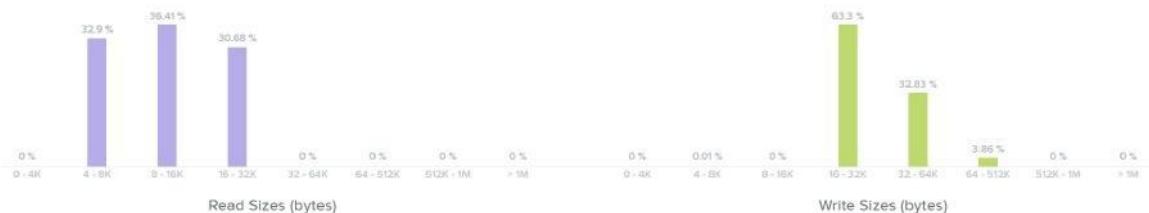


圖. I/O度量-I/O規模長條圖

在這裡可以看到讀 I/O 範圍從 4K 到 32K 大小：

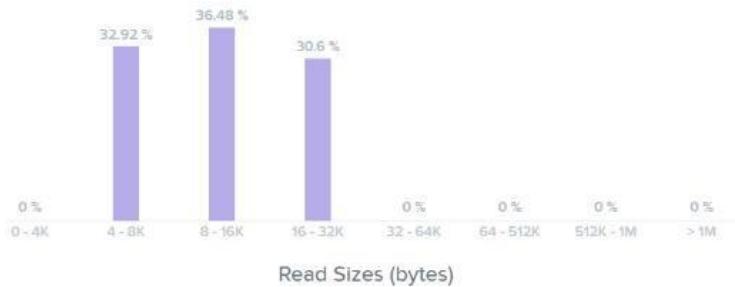


圖. I/O度量-讀I/O規模長條圖

在這裡可以看到寫 I/O 範圍從 16K 到 64K，以及一些到 512K 大小：

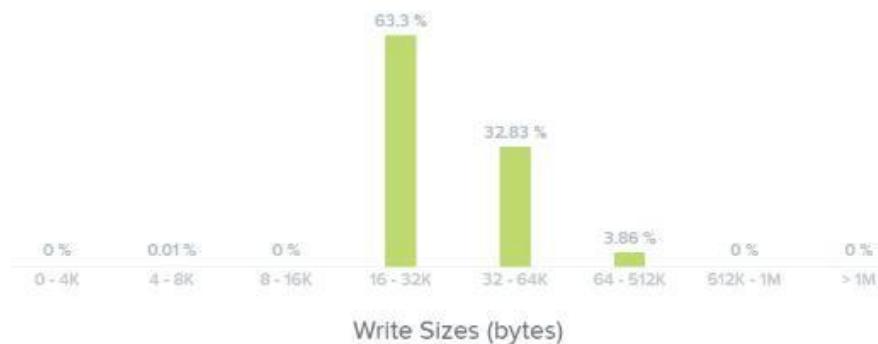


圖. I/O度量-寫I/O規模長條圖

專家提示

如果看到延遲出現一個尖峰，首先要檢查的是 I/O 大小。較大的 I/O (64K 到 1MB) 通常比小的 I/O 有更高的延遲 (4K 到 32K)。

下一節顯示為讀寫 I/O 延遲的長條圖

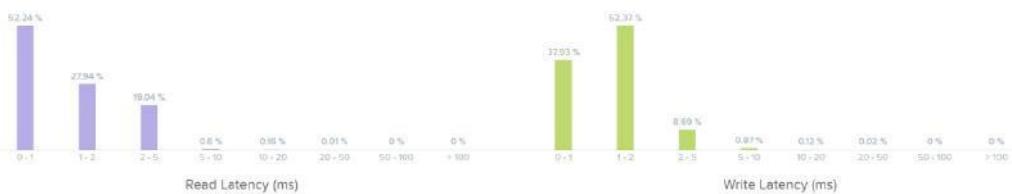


圖. I/O度量-延遲長條圖

通過看讀延遲長條圖中我們可以看到大多數讀 I/O 在亞毫秒級別（小於 1ms），部分是 2-5 毫秒。



圖. I/O度量-讀延遲長條圖

看看下面的“讀I/O的源頭”，可以看到大部分的I/O操作都是從SSD層提供的

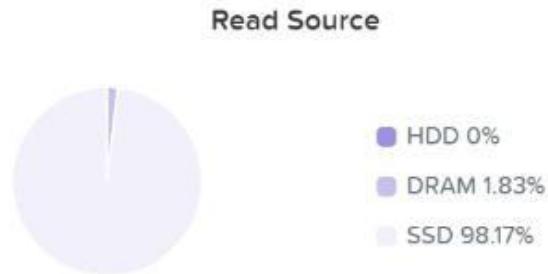


圖. I/O度量-讀的源頭SSD

當資料被讀取時，它將被即時獲取到統一的快取記憶體中（DRAM+SSD）（查看“I/O路徑和緩存”章節以瞭解更多資訊）。

在這裡，我們可以看到資料已被放到緩存中，現在正在從DRAM提供服務

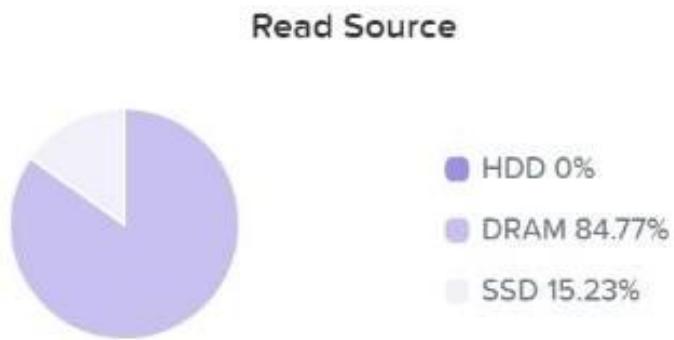


圖. I/O度量-讀的源頭DRAM

現在可以看到，基本上我們所有的讀I/O都是亞毫秒級 MS延遲 (<1ms)

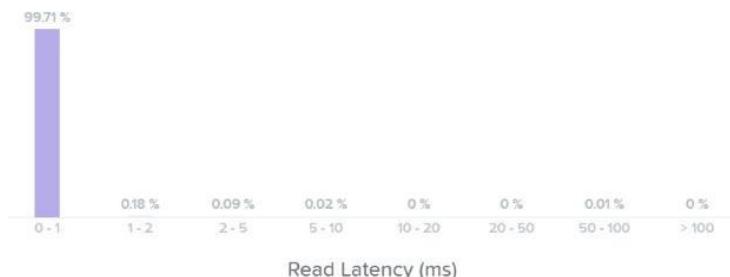


圖. I/O度量-讀延遲長條圖

在這裡可以看到，大多數的寫I / O是小於1-2毫秒的：



圖. I/O度量-寫延遲長條圖

專家提示

如果看到讀延遲出現了一個尖峰，並且I/O規模不大，請檢查正在讀取的I/O的位置
來自硬碟的任何初次讀取會比DRAM緩存有更長的延遲/等待時間

然而，一旦在緩存中，所有後續讀取都將命中DRAM，在延遲方面可以看到改進。

最後這部分顯示I/O類型與隨機讀寫和順序讀寫比例：



圖. I/O度量-隨機讀寫 vs. 持續讀寫

通常，I/O模式隨著應用程式或工作負載而變化（如VDI主要是隨機I/O，而Hadoop將主要順序I/O）。其他工作負載將是兩者的混合。例如，資料庫在插入或一些查詢操作時是隨機I/O，而在資料抽取過程中是持續I/O。

3.4.6 容量規劃

查看詳細的容量規劃細節，可以通過在 Prism Central 的“cluster runway”中點擊特定的集群來查看容量規劃的具體資訊。



圖. Prism Central - 容量規劃

這張視圖提供了關於集群使用趨勢的具體資訊，並確定了最吃緊的資源情況（限制資源）。你也可以從中瞭解到佔用資源最多的那些虛擬機器，並得到可能的選項。譬如一些釋放資源的建議或擴展集群的節點類型推薦。

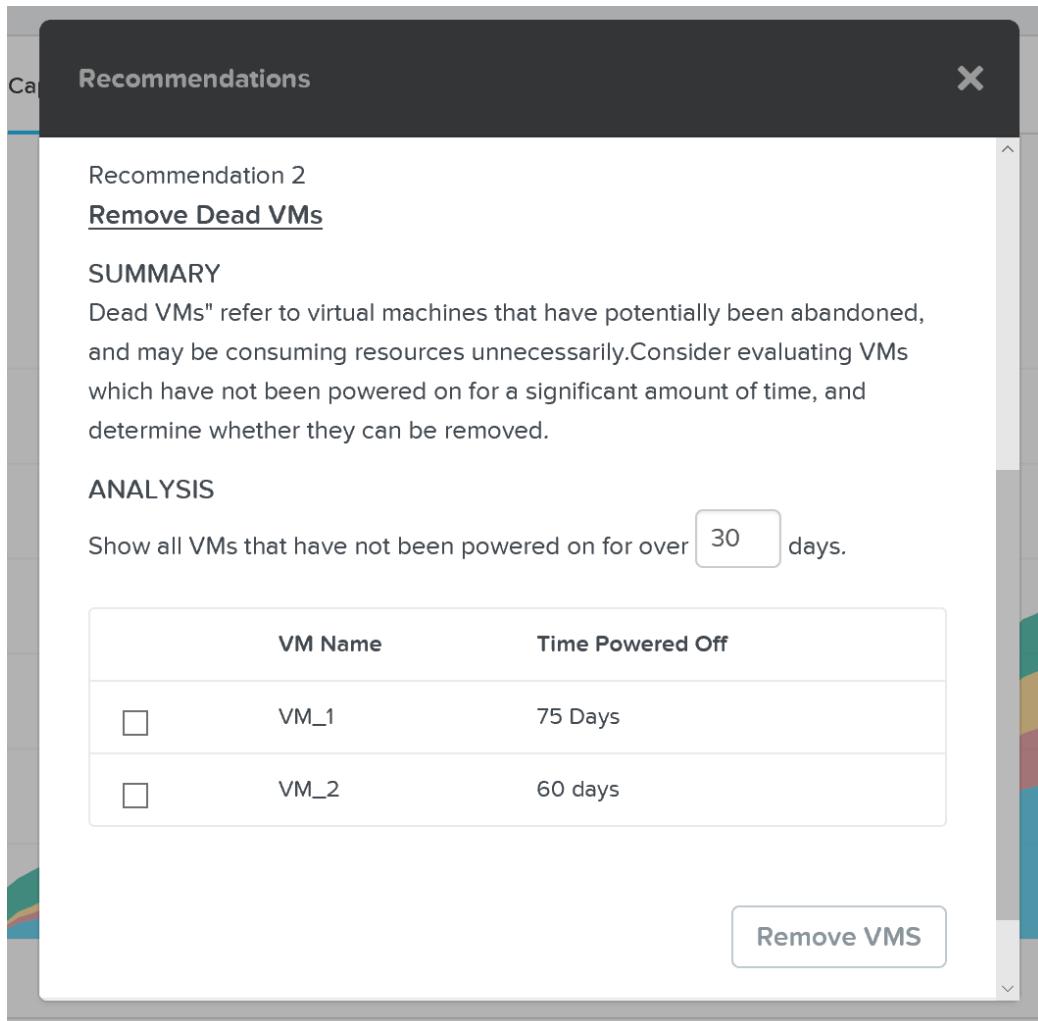


圖. Prism Central – 容量規劃 – 推薦建議

3.4.7 X-Play

當我們考慮自己的日常活動時，顯然是實現越多的自動化越好。我們在日常生活中經常使用常規程式，而科技使我們能夠在其他領域也做到這一點。Prism Pro X-Play允許我們通過 Prism 自動化一組常見的運維活動。但是，在深入研究產品之前，讓我們先介紹一下我們要做的事情。

事件以下列方式驅動的自動化工作：

事件→邏輯→行動

在這種情況下，發生某種事件（或級聯事件），會觸發一系列動作或一組動作。IFTTT就是一個很好的例子，它接受一個事件，應用一些邏輯（提示“if this then that”的縮寫），然後執行一些操作。



例如，當我們離開家時，關掉燈。如果我們可以對事件進行程式設計（例如，不在家中/離開設備），觸發系統自動關閉所有燈，那麼我們的生活就會變得更加簡單。我自己在家裡使用它，它使生活變得更加輕鬆，並使我能夠專注於其他更重要的活動。如果將此與我們的 IT 運營活動進行比較，我們將看到類似的模式。發生事件（例如，虛擬機器需要更多磁碟空間），然後我們執行一系列操作（例如，創建工單，添加存儲，關閉工單等）。這些重複性活動是自動化可以增加價值並使我們專注于更有益的活動的完美示例。

使用 X-Play，我們可以採取一系列事件/警報，並允許系統攔截這些事件/警報並執行一系列動作。

首先，請轉到 Prism Central 中“操作”下的“播放”部分：

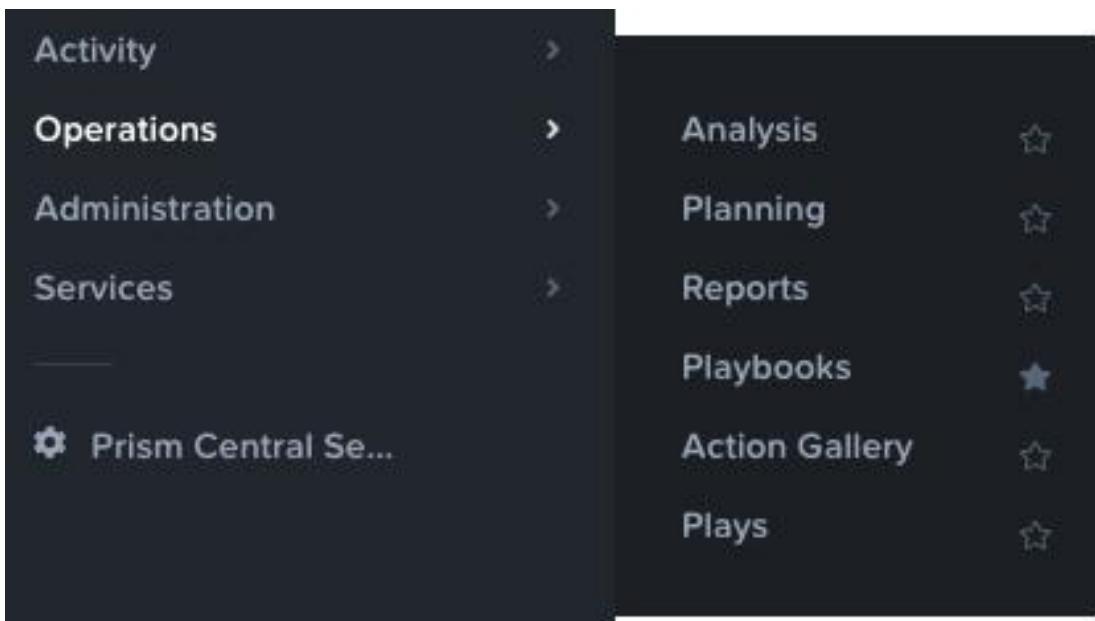


圖. X-Play – 導航頁面



這將啟動 X-Play 主頁：

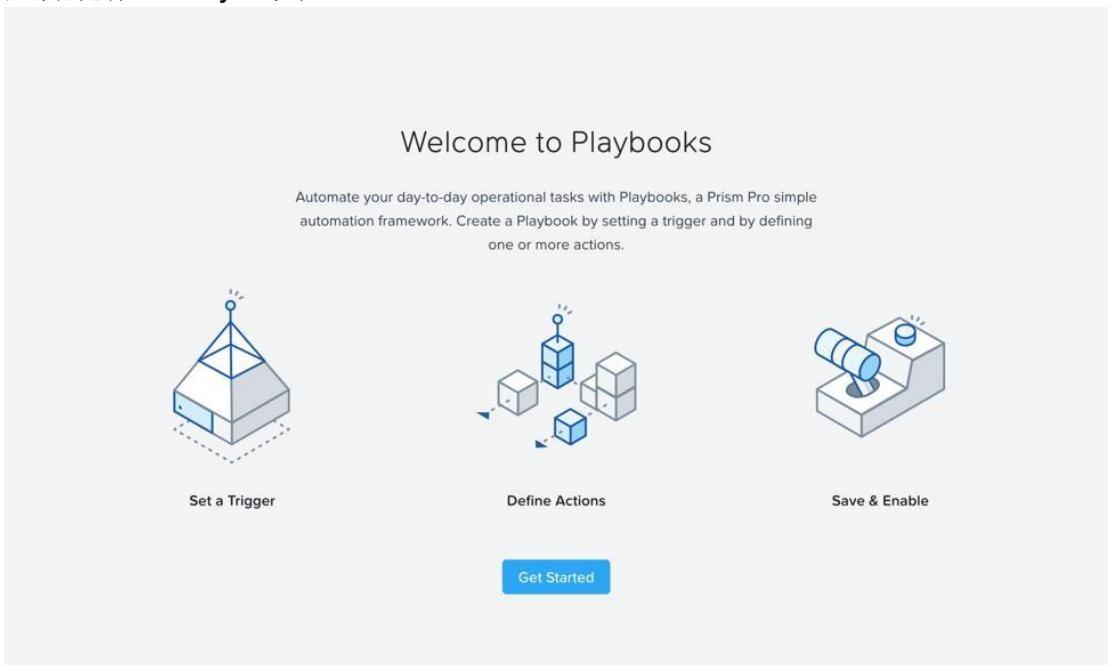


圖. X-Play – 劇本

在這裡，你可以通過定義觸發器來創建新的劇本：

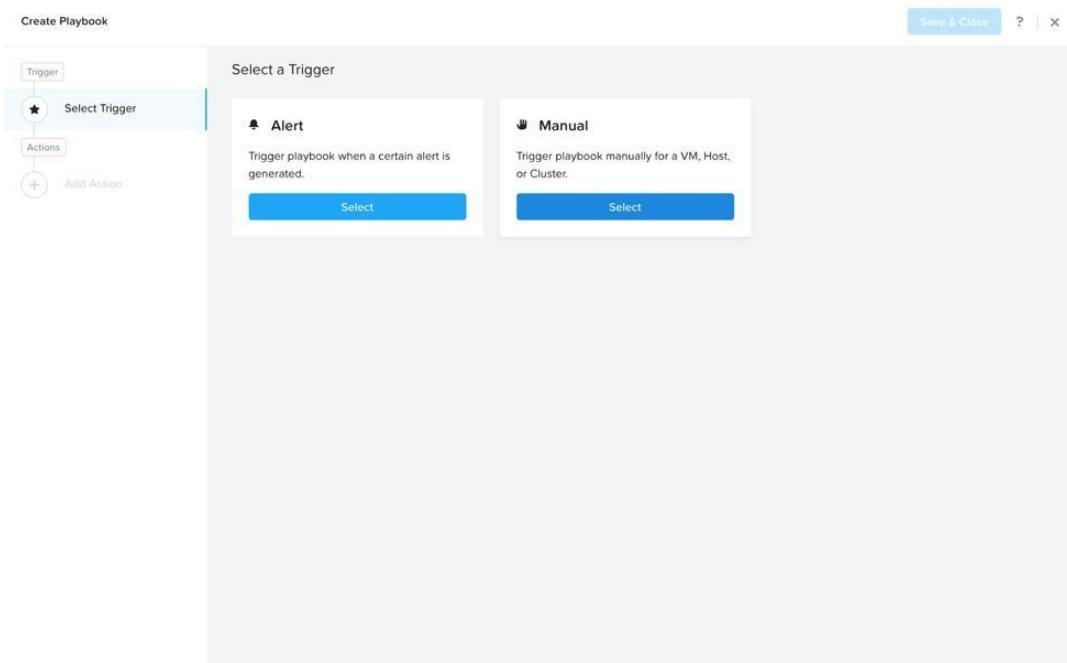


圖. X-Play – 觸發器

下面顯示了一個基於使用者自訂告警的觸發器示例：

Alert

Change Trigger

Alert Policy

AppType:Earth_Stack - VM Memory Usage

Filter Criteria

The selected policy has the following criteria defined. You could refine the criteria further to target the right alerts and right entities.

Severity

Critical

Target VM

- All Applicable VMs
- Specify VMs

圖. X-Play – 觸發器 – 用戶自訂告警

當定義好觸發器後，你現在可以指定一系列操作。下面顯示了一些示例操作：

The screenshot shows the 'Create Playbook' interface under the 'Alert' trigger. On the left, there's a flowchart editor with nodes for 'Trigger', 'Actions', and 'New Action'. The main area is titled 'Select an Action' and displays a grid of 16 operations categorized by color-coded icons:

- Blue Row:**
 - Acknowledge Alert
 - Email
 - Generate Forecast Report
 - IP Address Powershell
 - IP Address SSH
- Green Row:**
 - Power Off VM
 - Power On VM
 - Resolve Alert
 - REST API
 - Slack
- Yellow Row:**
 - VM Add CPU
 - VM Add Memory
 - VM Powershell
 - VM Reduce CPU
 - VM Reduce Memory
- Red Row:**
 - VM Snapshot
 - VM SSH
 - Wait for Some Time
 - Wait until Day of Month
 - Wait until Day of Week

Each action card includes a brief description, a 'Select' button, and a small 'Action Type' label (e.g., Alert Action, Communication Action, Report Action, Execution Action, System). A search bar and a 'All Actions' link are at the top right of the grid.

圖. X-Play -操作

之後，您輸入操作的詳細資訊，這列舉了一個 REST API 調用的列子：



REST API

Change Action

Settings

On Action Failure: Stop

Method

POST

URL

https://myurl.nutanix.com/myEndpoint



Username

foo

Password

...



Request Body

```
{  
  "customTicket": {  
    "entity": "[trigger[0].source_entity_info.name]",  
    "eventTime": "[trigger[0].creation_time]",  
    "eventType": "[trigger[0].alert_entity_info.name]",  
  }  
}
```

Parameters

Request Headers

Content-Type: application/json

圖. X-Play - REST Action 例子

REST API 操作和外部系統

X-Play 提供了許多預設操作，例如發送電子郵件，發送 slack 消息以及執行 REST API 調用等其他操作。當我們考慮對接 CMDB 或其他工單/自動化工具之類的外部系統時，這一點至關重要。通過使用 REST API 操作，我們可以與那些介面進行交互以創建/解決工單，啟動其他工作流程等。這是一個非常強大的選項，它可以使所有系統保持同步。

對於實體/事件的特定詳細資訊，你可以使用“參數”這個變數，該變數將為您提供有關事件，實體和其他內容的詳細資訊：



圖. X-Play - 操作的參數

完成後，你可以保存你的播放，它將按照定義開始執行。
下面顯示了一個執行多個操作的 Play 示例：

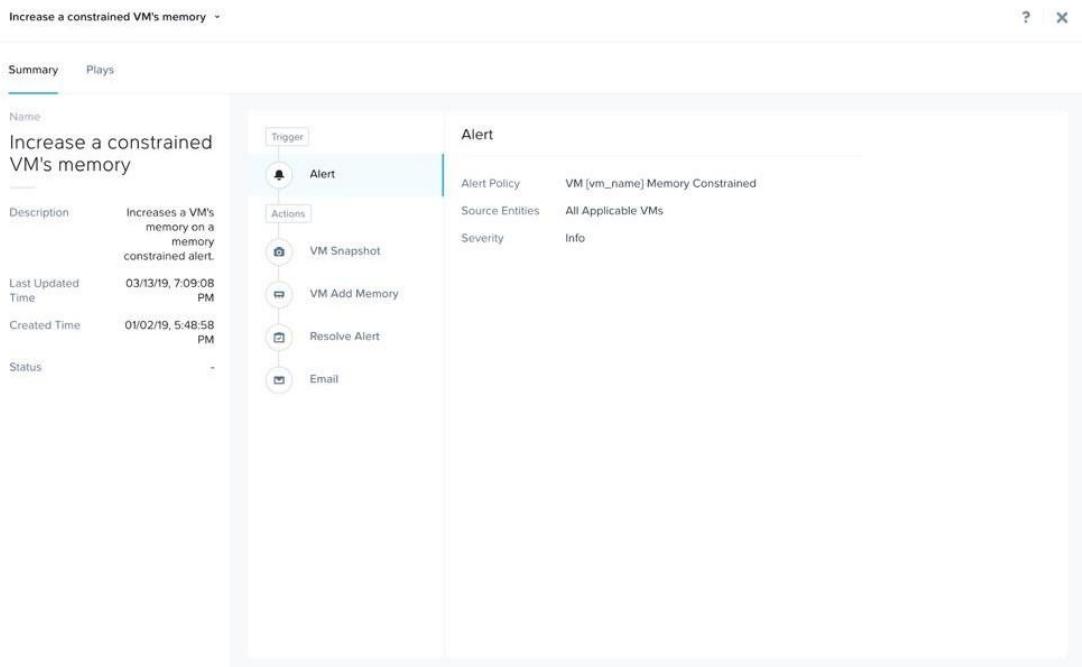


圖. X-Play – Playbook 示例

Play 標籤框裡會顯示 Play 的執行時間和狀態：



Summary Plays

1 - 20 of 203 < >

Playbook	Trigger	Status	Occurrence Time	Last Updated Time
Earth_Stack: Memory Alert	Alert	Succeeded	03/14/20, 08:52:50 AM	03/14/20, 08:52:50 AM
Earth_Stack: Memory Alert	Alert	Succeeded	03/14/20, 08:52:50 AM	03/14/20, 08:52:50 AM

圖.X-Play – 執行

記住，自動化所有事情！

3.5 APIs 介面

基於 HTML5 的使用者介面是讓 Prism 成為易於使用的管理工具的關鍵部分。同時，另一項核心功能是用於自動化的 API。所有 Prism 使用者介面提供的功能都可以通過 REST API 程式設計接口供外部程式調用。這使得用戶或是合作夥伴可以實現自動化部署、協力廠商工具集成甚至是創建他們自己的使用者介面。

在動態的、軟體定義的環境中，Nutanix 提供了一組/系列介面，讓程式調用變得非常簡單。以下是主要的介面：

- REST API
- CLI - ACLI & NCLI
- Scripting interfaces

developer.nutanix.com

要瞭解有關 API 的更多資訊並查看示例代碼，請務必查 developer.nutanix.com

作為這其中的關鍵部分，REST API 可調用 Prism 的每一個功能和資料點，讓工作流/自動化工具可以輕易的驅動 Nutanix 做出反應。可與 Nutanix 集成的自動化工具包括 Saltstack，Puppet，vRealize Operations，System Center Orchestrator 和 Ansible 等，都可以非常方便的創建構建在 Nutanix 上的客戶工作流。這當然也意味著任何協力廠商開發人員都能通過 REST 創建他們自己的使用者介面或從 Nutanix 平臺提取資料。

下圖展示了 Nutanix REST API 流覽器的一小部分，以表明開發者是如何調用這些 API 介面並得到其所希望的資料格式：



圖. Prism REST API 流覽器

可以展開操作來顯示 REST 調用的具體資訊和例子：

This screenshot shows a detailed view of the Prism REST API for the `/cluster/public_keys/{name}` endpoint. It includes:

- GET /cluster/public_keys/{name}**: Description: Get a Public Key. Parameters: name (required). Data Type: string.
- DELETE /cluster/public_keys/{name}**: Description: Delete a Public Key.
- GET /cluster/**: Description: Get Cluster details.

Implementation Notes: Get a Public Key with the specified name.

Parameters table:

Parameter	Value	Description	Data Type
name	(required)	Name of the Public Key	string

Error Status Codes table:

HTTP Status Code	Reason
500	Any internal exception while performing this operation

A "Try it out!" button is also present.

圖. Prism REST API 調用實例

API 驗證方案

對於版本 4.5.x 用戶端和 HTTP 調用的驗證都通過使用 HTTPS 的基本驗證方式來實現。

3.5.1 ACLI

Acropolis CLI (ACLI) 是用來管理 Nutanix 產品中 Acropolis 那部分功能的命令列介面。這些功能從 4.1.2 之後被開放。

說明：所有這些操作都可以通過 HTML5 圖形化使用者介面和 REST API 完成。我這裡用的命令只是作為我用來實現自動化腳本的一部分。

進入 ACLI Shell

描述: 進入 ACLI shell (可從任何 CVM 運行)。

Acli



或者

描述: 通過 Linux shell 執行 ACCLI 命令

ACLI <Command>

ACLI 的回應以 json 格式輸出

描述: 以 json 格式輸出 acl 命令的結果

Acli -o json

列出主機

描述: 列出集群中的 Acropolis 節點

host.list

創建網路

描述: 根據 VLAN 創建網路

net.create <TYPE>.<ID>[.<VSWITCH>] ip_config=<A.B.C.D>/<NN>

Example: net.create vlan.133 ip_config=10.1.1.1/24

列出網路 v

描述: 列出網路

net.list

創建 DHCP 範圍

描述: 創建 DHCP 範圍

net.add_dhcp_pool <NET NAME> start=<START IP A.B.C.D> end=<END IP W.X.Y.Z>

說明: 如果在創建網路時沒有指定 Acropolis DHCP 伺服器的位址, 那麼 .254 被保留並用作 Acropolis DHCP 伺服器地址。



Example: net.

獲得當前網路的詳細資訊

描述：獲得一個網路中的虛擬機器及其名字、UUID、MAC 位址及 IP 位址等資訊

```
net.list_vms <NET NAME>
```

Example: net.list_vms vlan.133

為網路配置 DHCP 和 DNS 伺服器

描述：設置 DHCP 和 DNS

```
t.update_dhcp_dns <NET NAME> servers=<COMMA SEPARATED DNS IPs> domains=<COMMA  
SEPARATED DOMAINS>
```

Example: net.set_dhcp_dns vlan.100 servers=10.1.1.1,10.1.1.2

創建虛擬機器

描述：創建虛擬機器

```
vm.create <COMMA SEPARATED VM NAMES> memory=<NUM MEM MB> num_vcpus=<NUM VCPU>  
num_cores_per_vcpu=<NUM CORES> ha_priority=<PRIORITY INT>
```

Example: vm.create testVM memory=2G num_vcpus=2

批量創建虛擬機器

描述：批量創建虛擬機器

```
vm.create <CLONE PREFIX>[<STARTING INT>..<END INT>] memory=<NUM MEM MB>  
num_vcpus=<NUM VCPU> num_cores_per_vcpu=<NUM CORES> ha_priority=<PRIORITY INT>
```

Example: vm.create testVM[000..999] memory=2G num_vcpus=2

基於已有虛擬機器克隆

描述：創建已有虛擬機器的克隆

```
vm.clone <CLONE NAME(S)> clone_from_vm=<SOURCE VM NAME>
```



```
Example vm.clone testClone clone_from_vm=MYBASEVM
```

基於已有虛擬機器批量克隆

描述：批量創建已有虛擬機器的克隆

```
vm.clone <CLONE PREFIX>[<STARTING INT>..<END INT>] clone_from_vm=<SOURCE VM NAME>
```

```
Example vm.clone testClone[001..999] clone_from_vm=MYBASEVM
```

創建磁片並添加到虛擬機器

```
# Description: Create disk for OS
```

```
vm.disk_create <VM NAME> create_size=<Size and qualifier, e.g. 500G> container=<CONTAINER NAME>
```

```
class="codetext">Example vm.disk_create testVM create_size=500G container=default
```

給虛擬機器添加網卡

描述：創建並添加網卡

```
vm.nic_create <VM NAME> network=<NETWORK NAME> model=<MODEL>
```

```
Example vm.nic_create testVM network=vlan.100
```

設置虛擬機器的啟動磁片

描述：設置啟動設備

通過磁片 ID 設置從指定的磁片啟動

```
vm.update_boot_device <VM NAME> disk_addr=<DISK BUS>
```

```
Example vm.update_boot_device testVM disk_addr=scsi.0
```

設置虛擬機器的啟動光

碟機設置從光碟機啟動

```
vm.update_boot_device <VM NAME> disk_addr=<CDROM BUS>
```



```
Example: vm.update_boot_device testVM disk_addr=ide.0
```

掛載 ISO 到光碟機

描述：掛載 ISO 到虛擬機器的光碟機

步驟：

1. 上載 ISOs 到 container
2. 為用戶端 IP 位址設置訪問白名單
3. 上載 ISOs 以共用基

於 ISO 創建光碟機設備

```
vm.disk_create <VM NAME> clone_nfs_file=<PATH TO ISO> cdrom=true
```

```
Example: vm.disk_create testVM clone_nfs_file=/default/ISOs/myfile.
```

如果光碟機已經創建則直接掛載

```
vm.disk_update <VM NAME> <CDROM BUS> clone_nfs_file=<PATH TO ISO>
```

```
Example: vm disk_update atestVM1 ide.0 clone_nfs_file=/default/ISOs/myfile.iso
```

從光碟機卸載 ISO

描述：從光碟機移除 ISO

```
vm.disk_update <VM NAME> <CDROM BUS> empty=true
```

開啟虛擬機器

描述：開啟虛擬機器

```
vm.on <VM NAME(S)>
```

```
Example: vm.on testVM
```

開啟所有虛擬機器



Example: vm.on *

開啟一段編號範圍的虛擬機器

Example: vm.on testVM[01..99]

3.5.2 NCLI

說明：所有這些操作都可以通過 HTML5 圖形化使用者介面和 REST API 完成。我這裡用的命令只是作為我用來實現自動化腳本的一部分。.

添加子網到 NFS 白名單

描述：添加特定的子網到 NFS 白名單

ncli cluster add-to-nfs-whitelist ip-subnet-masks=10.2.0.0/255.255.0.0

顯示 Nutanix 版本

描述：顯示當前 Nutanix 軟體的版本

ncli cluster version

顯示 NCLI 的隱藏選項

描述：顯示 ncli 的隱藏命令/選項

ncli helpsys listall hidden=true [detailed=false|true]

列出存儲資源池

描述：顯示已存在的存儲資源池

ncli sp ls

列出存儲容器

描述：列出已有的容器

ncli ctr ls

創建存儲容器



描述：創建一個新的容器

```
ncli ctr create name=<NAME> sp-name=<SP NAME>
```

列出虛擬機器

描述：顯示已存在的虛擬機器

```
ncli vm ls
```

列出公共金鑰

描述：列出已存在公共金鑰

```
ncli cluster list-public-keys
```

添加公共金鑰

描述：為集群訪問添加公共金鑰

通過 SCP 傳輸公共金鑰到 CVM

添加公共金鑰到集群

```
ncli cluster add-public-key name=myPK file-path=~/mykey.pub
```

移除公共金鑰

描述：移除集群訪問的公共金鑰

```
ncli cluster remove-public-keys name=myPK
```

創建保護域

描述：創建一個保護域

```
ncli pd create name=<NAME>
```

創建遠程網站

描述：為資料複製創建遠端網站



```
ncli remote-site create name=<NAME> address-list=<Remote Cluster IP>
```

為存儲容器內的所有虛擬機器創建保護域

描述：保護特定存儲容器中的所有虛擬機器

```
ncli pd protect name=<PD NAME> ctr-id=<Container ID> cg-name=<NAME>
```

為特定虛擬機器創建保護域

描述：保護特定虛擬機器

```
ncli pd protect name=<PD NAME> vm-names=<VM Name(s)> cg-name=<NAME>
```

為 DSF 檔（亦稱作 vDisk）創建保護域

描述：保護特定的 DSF 檔

```
ncli pd protect name=<PD NAME> files=<File Name(s)> cg-name=<NAME>
```

創建保護域的快照

描述：為保護域創建一次性的快照

```
ncli pd add-one-time-snapshot name=<PD NAME> retention-time=<seconds>
```

創建快照和同步到遠端網站的複製計畫

描述：創建一個持續的快照計畫，同步資料到遠端網站

```
ncli pd set-schedule name=<PD NAME> interval=<seconds> retention-policy=<POLICY> remote-sites=<REMOTE SITE NAME>
```

列出複製狀態

描述：監控資料複製狀態

```
ncli pd list-replication-status
```

遷移保護域到遠程網站



描述：故障切換一個保護域到遠端網站

```
ncli pd migrate name=<PD NAME> remote-site=<REMOTE SITE NAME>
```

啟動保護域

描述：啟動遠程網站的保護域

```
ncli pd activate name=<PD NAME>
```

開啟 DSF 的 Shadow Clones

描述：開啟 DSF 的 Shadow Clone 功能

```
ncli cluster edit-params enable-shadow-clones=true
```

打開 vDisk 的去重

描述：為特定 vDisk 開啟指紋識別並啟用去重

```
ncli vdisk edit name=<VDISK NAME> fingerprint-on-write=<true/false> on-disk-dedup=<true/false>
```

檢查集群可恢復性狀態

```
Node status
```

```
ncli cluster get-domain-fault-tolerance-status type=node
```

```
# Block status
```

```
ncli cluster get-domain-fault-tolerance-status type=rackable_unit
```

3.5.3 PowerShell CMDlets

下面將講述 Nutanix 對 PowerShell CMDlets 命令列支持，包括如何使用和一些關於 Windows PowerShell 的背景知識。

基礎

Windows PowerShell 是構建在.net 框架上的強大的 shell 和指令碼語言。它是非常易用的程式設計語言，直觀且交互性好。PowerShell 中有一些關鍵結構和元件：



CMDlets

CMDlets 是一組執行特定操作的命令或.NET 類。它們符合 Getter/Setter 方法論，通常使用<動詞>-<名詞>的操作結構。例如：Get-Process, Set-Partition 等。

管道 (Piping) 和流水線 (Pipelining)

管道是 PowerShell 中的重要概念（類似于 Linux 中的用法），如果能正確使用，能使很多事變得簡單。通過管道，你基本上可以把一個流水線操作的輸出導向到作為下一個流水線操作的輸入。流水線可以根據需要變長。一個非常簡單的例子就是獲得當前進程，找到匹配特定特徵或篩檢程式的進程，然後對它們進行排序：

```
Get-Service | where {$_.Status -eq "Running"} | Sort-Object
```

管道也可以被用在 `for-each` 迴圈中，例如：

```
# For each item in my array  
$myArray | %{  
    # Do something  
}
```

關鍵對象類型

下面是 PowerShell 中一些關鍵物件類型。你可以通過`.GetType()`方法非常容易的得到一個物件的類型，比如：`$someVariable.GetType()`將會返回特定物件的類型。

變數

```
myVariable = "foo"
```

說明：你可以為變數賦值為一系列命令或管道的輸出：

```
$myVar2 = (Get-Process | where {$_.Status -eq "Running" })
```

在這個例子中，將先運算括弧中的命令，然後把其賦值給變數。

陣列



```
$myArray = @("Value","Value")
```

說明：你可以得到陣列、雜湊表或自訂物件的陣列。

雜湊表

```
$myHash = @{"Key" = "Value";"Key" = "Value"}
```

有用的命令

獲得特定 **CMDlet** 命令的說明資訊（類似於 Linux 中的 man）

```
Get-Help <CMDlet Name>
```

Example: Get-Help Get-Process

列出一個命令或物件的屬性和方法

```
<Some expression or object> | Get-Member
```

Example: \$someObject | Get-Member

核心 **Nutanix CMDlets** 和使用方法

下載 Nutanix CMDlets 安裝器。Nutanix CMDlets 可以通過 Prism (4.0.1 以後版本) 使用者介面直接下載，在右上角的下拉清單中可以找到。

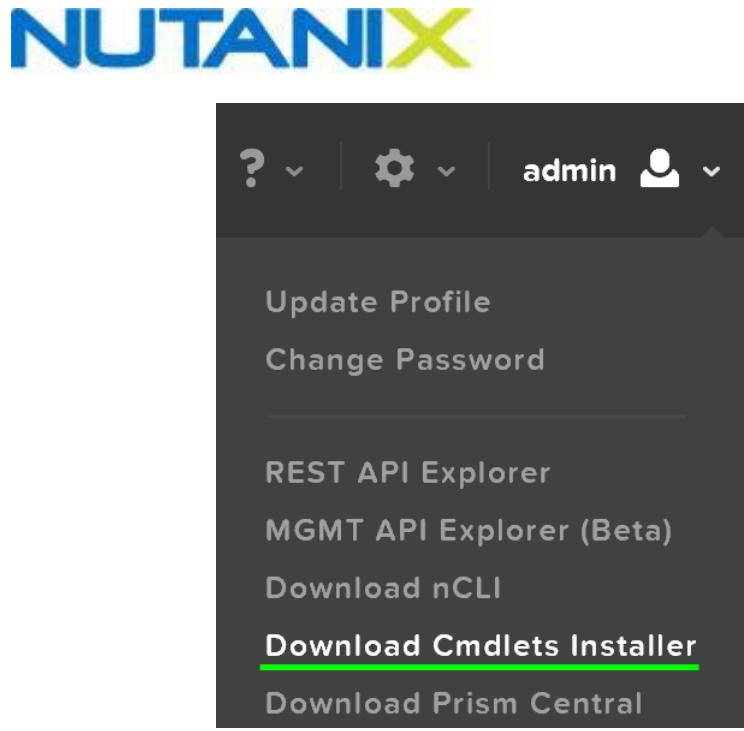


圖. Prism CMDlets 安裝器連結

裝載 Nutanix Snap-in

檢查 snap-in 是否被裝載，如沒有請裝載

```
if ( (Get-PSSnapin -Name NutanixCmdletsPSSnapin -ErrorAction SilentlyContinue) -eq $null )  
{  
    Add-PsSnapin NutanixCmdletsPSSnapin  
}
```

列出 Nutanix CMDlets

```
Get-Command | Where-Object{$_.PSSnapin.Name -eq "NutanixCmdletsPSSnapin"}
```

連接到一個 Acropolis 集群

```
ct-NutanixCluster -Server $server -UserName "myuser" -Password (Read-Host
```

```
"Password: " -AsSecureString) -AcceptInvalidSSLCerts
```

獲得滿足搜索條件的虛擬機器



設置變數

```
$searchString = "myVM"  
$vms = Get-NTNXVM | where {$_.vmName -match $searchString}
```

互動式

```
Get-NTNXVM | where {$_.vmName -match "myString"}
```

互動式的格式化輸出

```
Get-NTNXVM | where {$_.vmName -match "myString"} | ft
```

獲得 Nutanix vDisks

變數方式

```
$vdisks = Get-NTNXVDisk
```

互動式

```
Get-NTNXVDisk
```

互動式的格式化輸出

```
Get-NTNXVDisk | ft
```

獲得 **Nutanix** 存儲容器

設置變數

```
$containers = Get-NTNXContainer
```

互動式

```
Get-NTNXContainer
```

互動式的格式化輸出

```
Get-NTNXContainer
```



獲得 Nutanix 保護域

設置變數

```
pds = Get-NTNXProtectionDomain
```

互動式

```
Get-NTNXProtectionDomain
```

互動式的格式化輸出

```
Get-NTNXProtectionDomain | ft
```

獲得 Nutanix 一致性組

設置變數

```
$cgs = Get-NTNXProtectionDomainConsistencyGroup
```

互動式

```
Get-NTNXProtectionDomainConsistencyGroup
```

互動式的格式化輸出

```
Get-NTNXProtectionDomainConsistencyGroup | ft
```

資源和腳本：

- Nutanix Github - <https://github.com/nutanix/Automation>
- Manually Fingerprint vDisks - <http://bit.ly/1syOqch>
- vDisk Report - <http://bit.ly/1r34MIT>
- Protection Domain Report - <http://bit.ly/1r34MIT>
- Ordered PD Restore - <http://bit.ly/1pyolrb>

注意：

上面的一些腳本沒有被維護，僅供參考。

你可以在 Nutanix Github 上找到更多腳本

glance image-show <IMAGE_ID>

4 第四部分： Acropolis

4.1 架構

Acropolis 作業系統（AOS）提供了平臺上運行工作負載和服務所用到的核心功能。包括但不限於存儲服務、升級等。

該圖中高亮顯示的圖像，展示了 Acropolis 在各個層面的概念性質：

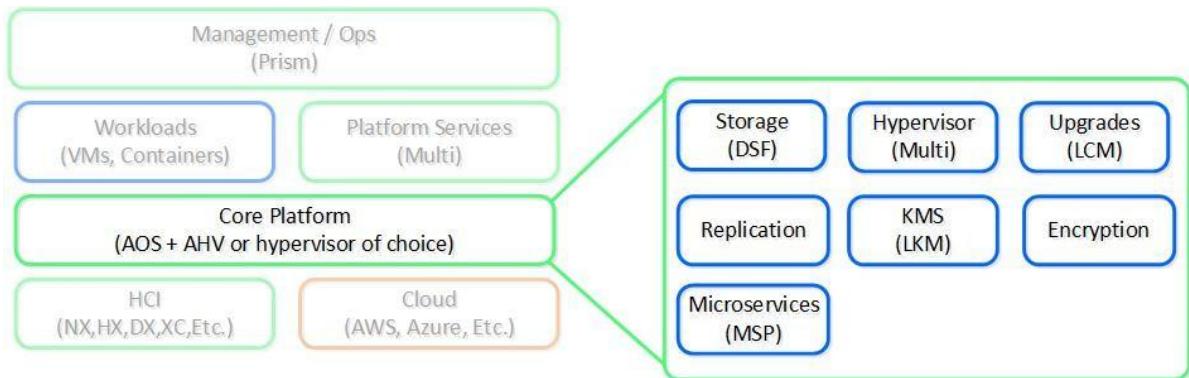


圖. Acropolis 高級架構

基於 Nutanix 的分散式特性，我們將其擴展到虛擬化和資源管理領域。Acropolis 是一個後端服務，支援工作負載和資源管理、配置和操作。它的目標是從運行的工作負載中抽象出便利的資源(例如，系統管理程式、內部環境、雲等等)，同時提供一個單一的“平臺”進行操作。

這使得工作負載能夠在管理程式、雲提供商和平臺之間無縫移動。

支援的虛擬機器管理程式

從 4.7 開始，支持 AHV 和 ESXi 虛擬機器管理程式，但是未來可能會擴大對 Hypervisor 的支持。所有卷 API 和唯讀操作仍然受支持。

4.1.1 Acropolis 服務



集群內的每個 CVM 上都會運行一個 Acropolis 服務，其中一個會被選舉為首選 Acropolis，負責任務調度，任務執行，IPAM（IP 位址管理）等，其餘的均為從屬 Acropolis，類似於其他具有首選節點的元件，當集群中首選 Acropolis 發生故障時，就會在餘下的 Acropolis 服務中選舉出一個新的首選 Acropolis。

Acropolis 服務的角色分類如下：

主 Acropolis

- 任務調度&執行
- 統計資訊收集/發佈
- 網路控制器（針對 Hypervisor）
- VNC 代理（針對 Hypervisor）
- HA（針對 Hypervisor）

從屬 Acropolis

- 統計資訊收集/發佈
- VNC 代理（針對 Hypervisor）

下圖展示了首選 Acropolis 與從屬 Acropolis 之間關係的概念視圖：

An Acropolis Master is elected per cluster and is responsible for task scheduling, HA, VNC proxy, etc.

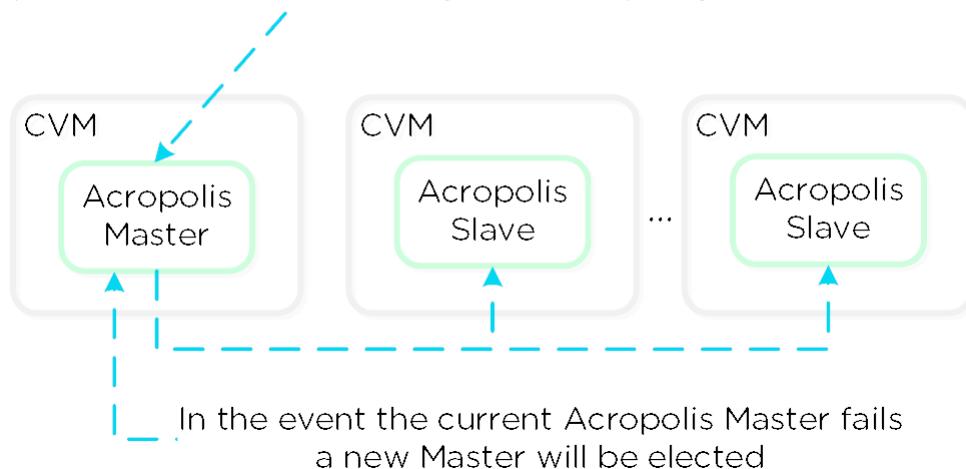


圖. Acropolis 服務

4.1.2 動態調度

有效地安排資源是確保資源有效利用的關鍵。Acropolis 動態調度增強了傳統調度方法需要依賴計算資源(CPU/MEM)來做出放置決策的調度方法。它充分利用了計算，存儲和其他資源來驅動 VMs 和卷(ABS)放置決策。這確保了有效的資源消耗，從而獲得最佳的最終使用者性能體驗。

資源調度由兩個關鍵過程組成：

- 初始位置放置
虛擬機器在那台伺服器節點上開機
- 運行時優化
基於即時運行指標的工作負載移動

Acropolis 調度在虛擬機器初始位置放置時即開始生效。隨著在 Asterix 版本的發佈，Acropolis 動態調度在此基礎上擴展並提供即時的資源優化功能。

圖中顯示了調度架構的高級視圖：

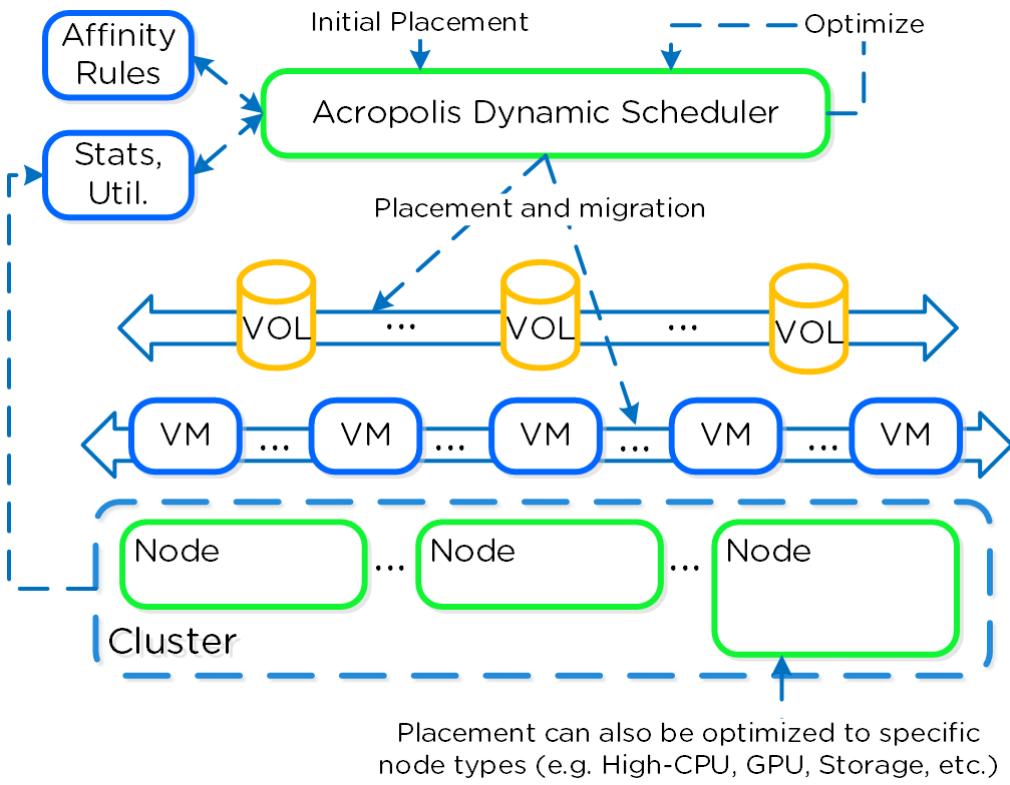


圖.Acropolis 動態調度

動態調度機制始終運行，從而優化放置位置，(目前每 15 分鐘| Gflag:
`lazan_anomaly_detection_period_secs`)。使用歷史利用數值和平滑演算法計算預估需求。這個預估的需求考慮到動態因素，這保證了突然的峰值不會擾亂結果。

獨特的資源優化方法

審視現有的調度/優化平臺(VMware DRS, Microsoft PRO)時，它們都只關注于在集群資源之間均勻地平衡工作負載，需要注意的是，如何消除激進的資源不平衡，主要取決於配置(例如手動->沒有，保守->一些，激進->更多)。

例如，假設集群中有 3 個主機，每個主機分別使用 50%、5% 和 5%。典型的解決方案將嘗試重新平衡工作負載，使每個主機利用率達到 20%。但是真的有必要嗎？我們真正需要的是消除資源競爭，而不是消除不平衡。除非有資源競爭，否則“平衡”工作負載不會帶來任何好處。事實上，通過強迫不必要的移動，我們會導致額外的工作(例如，記憶體傳輸、緩存重新定位等)，所有這些都需要消耗資源。

Acropolis 動態調度機制是這樣的：它只會在預期發生資源競爭的情況下調用工作負載移動，而不是因為不平衡。注:Acropolis DSF 以不同的方式工作，以確保在整個集群中均勻分佈資料以消除熱點和加速重建。要瞭解更多 DSF，請查看“磁片平衡”部分。

在虛擬機器開機的時候。ADS 將平衡 VMs 在整個集群中的初始放置位置。



4.1.3 放置位置

放置位置取決於下面的幾個因素：

- 計算資源消耗

我們監控每個節點的計算利用率。如果一個節點的 CPU 分配違背了它的閾值(目前是主機 CPU 的 85% | Gflag: `lazan_host_cpu_usage_old_fraction`)，我們將從這主機上遷移出 VMs 來重新平衡工作負載。這裡要提到的一個關鍵問題是，只有在存在資源爭用時才會執行遷移。如果在節點之間(例如，3 節點 10% 和 1 節點 50%)之間的利用率有偏差，我們將不會執行遷移，因為這樣做沒有任何好處，除非有資源爭用。

- 存儲性能

作為一個超融合平臺，我們管理計算和存儲資源。調度程式將監視每個節點的 Stargate 進程利用率。當某些 Stargate(s)違反分配閾值(當前 CPU 分配給 Stargate 為 85% | Gflag: `lazan_stargate_cpu_usage_old_pct`)時，我們將在主機間遷移資源，以消除任何熱點。VMs 和 ABS 的卷都可以遷移以消除任何熱點的 Stargates 訪問。

- (反)關聯規則

關聯性或反關聯性約束決定了在環境中的某項資源在調度時需要基於其他特定資源所在的位置。例如在某些情況下，您希望 VMs 在相同的節點上運行，以獲得合規的許可。在這種情況下，VMs 將被綁定到同一個主機上。在其他情況下，您可能希望確保 VMs 在不同的節點上運行，以實現可用性。在這種情況下，虛擬機器將被反關聯性約束。

調度器將盡最大努力基於先前工作負載情況確保優化。該系統會對無謂的移動進行懲罰，以確保不會有太多的遷移發生。這是非常重要的，因為我們希望確保移動不會對工作負載產生任何負面影響。

在遷移之後，系統將判斷遷移的“有效性”，並瞭解實際的好處是什麼。通過這種智慧的學習模式可以自我優化，以確保任何遷移決策都有一個有效的依據。

4.2 安全與加密

自 Nutanix 平臺創建開始，安全就是 Nutanix 平臺的核心部分。 安全開發生命週期（**Security Development Lifecycle (SecDL)**）貫穿 Nutanix 開發過程的每一步。安全開發生命週期系統是從開發階段就考慮安全，而不是到最終使用者使用後根據客戶安全需求再開始考慮去“加固”平臺。

當我們考慮安全問題時，我們實際上是在努力實現三個核心目標(被恰當地稱為 CIA 三位一體):

1. 保密性
 - 通過防止未經授權的訪問來保護和保護資料
2. 完整性
 - 通過防止未經授權的更改，確保資料的一致性和準確性
3. 可用性
 - 確保獲授權使用者可通過彈性和冗餘訪問資料

這可以簡化為一個簡單的語句：允許使用者在不讓壞人進入的情況下完成他們的工作。當我們設計安全性時，我們需要看看幾個核心領域的利益，這些區域在下圖中突出顯示：

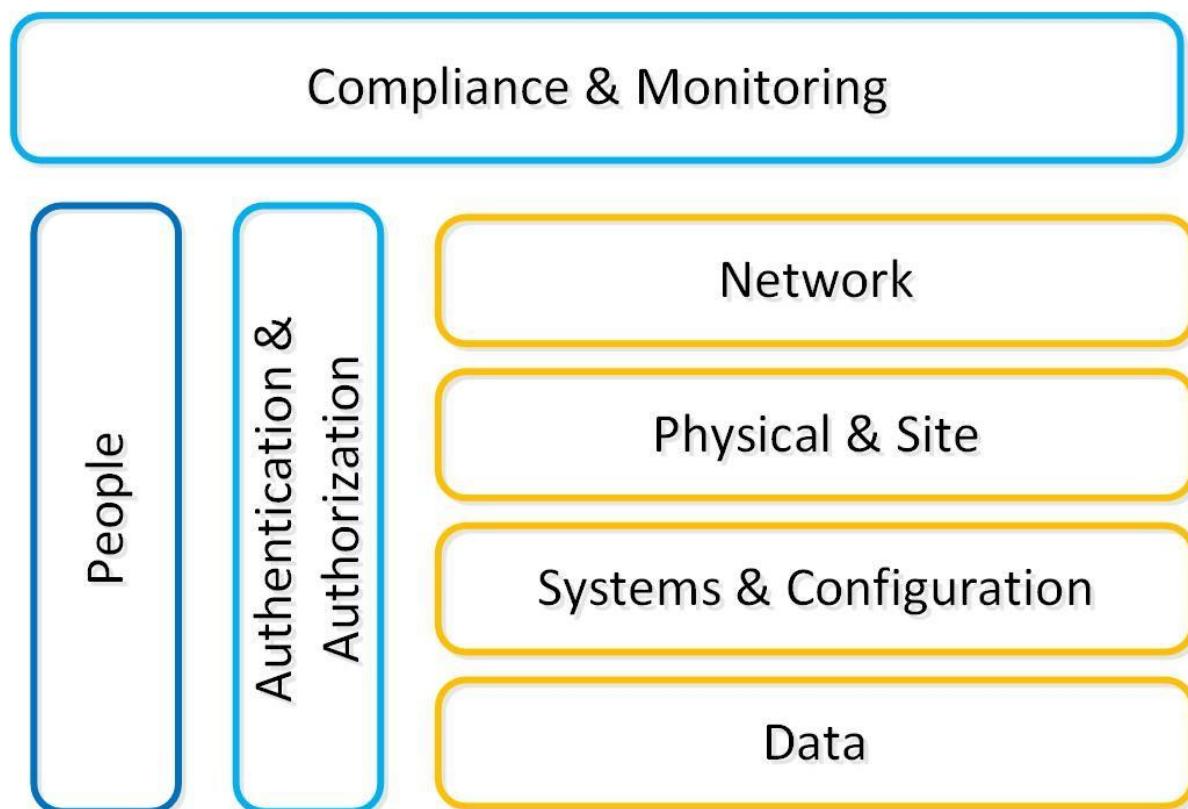


圖. 安全層面

在下面的章節中，我們將詳細分析先前圖表中的每一個部分。

系統和配置

快速一覽



- 修補並刪除已知的漏洞
- 強制使用強式密碼並刪除默認帳戶
- 配置許可權和使用者許可權
- 關閉未使用的埠/協定
- 使用自動化來確保基線

傳統上，人們使用一種稱為“加強”的方法來指代系統（OS+App）安全性。在這個過程中，您可以通過將某些內容配置為一個稱為基線的特定標準來保護系統。

國防部的 IT 組織（DISA）有一個樣本強化指南，他們稱之為 STIG（更多細節見下文 SCMA 部分）。這包括目錄許可權、使用者帳戶管理、密碼複雜性、防火牆和許多其他配置設置。

一旦一個系統被配置成這個標準，它就被認為是“安全的”，然而這僅僅是這個過程的開始。系統安全是在其整個生命週期中必須維護的東西。例如，為了確保滿足標準加固基線，應該配置使用自動化工具。這確保系統始終滿足您的基線“期望狀態”。

Nutanix 使用我們開發的一個稱為 SCMA 的工具來確保其 CVM 和 AHV 虛擬機器監控程序的安全，本節稍後將介紹這個工具。

數據

快速一覽

- 對資料的安全存取控制
- 始終備份
- 資料加密和安全金鑰

資料是任何業務的核心，可以說是公司最有價值的資產。考慮安全性時，我們需要確保資料的可訪問性，品質和避免洩露。

關於可訪問性的概念，我們始終需要訪問系統和資料來制定決策。最近一種稱為“勒索軟體”的攻擊方法通過加密資料，然後勒索用戶以獲取存取權限，從而對資料訪問構成威脅。可以通過多種方法來避免這種情況，這也突出了備份的重要性。

資料品質也是至關重要的一項，因為很多決定或行動都取決於它。例如，攻擊者可以訪問系統並下達惡意訂單或將貨物的收件地址改為他的地址。在這裡日誌和校驗對於確保資料不被篡改至關重要。



最後一點是我們如何保護或加固資料。通常是通過加密來完成的，如果沒有用於解密資料的金鑰，則加密過程將使資料變得無法使用。在這種情況下，如果有人要竊取加密的檔或磁片設備，則他們將無法訪問裡面的資料。

網路

快速一覽

- 劃分授信/不受信的網路
- 周邊和網段之間的防火牆
- 利用 **IDPS** 來檢測異常

網路是攻擊者用來獲取系統存取權限的典型通信媒介。其中包括周邊安全（例如外部防火牆）和內部入侵防護/檢測。

像任何好的設計一樣，網路也應該始終存在多個安全層級。我們需要將高安全性網路從受信任的網路中劃分出來，並從不受信任的網路（例如商業/ wifi 網路）中保護它們。我們要始終認為辦公室裡的網路是永遠不安全的。

通過擁有多級分層的網路，我們可以確保已經可以訪問我們不受信網路的人在想進入安全網路時會遇到更大的困難。在此過程中，好的 **IDPS** 系統可以檢測訪問異常或掃描工具，例如 **nmap**。

認證與授權

快速一覽

- 盡可能使用 MFA / 2FA
- 使用細化許可權

身份驗證就是根據可信賴的真實資料來源例如 **Active Directory** 或任何其他 **IDP**（提供身份驗證程式），對使用者身份進行身份驗證。**MFA**（多因素身份驗證）或**2FA** 這樣的工具增加了對用戶身份的額外保證。

一旦身份得到驗證，下一步就是確定他們被授權做什麼或可以訪問什麼；這就是授權部分。用戶 **foo** 被授權在 **bar** 上執行 **x, y**，在 **bas** 上執行 **y, z**。

合規與監控



快速一覽

- 合規是一項持續的活動
- 監控並查找異常

合規性通常是人們在查看某些認證（如 PCI、HIPA 等）時所指的東西。它進一步擴展到確保符合已設置的強化指南或標準。例如，STIG 是強化樣本的基線，但是每個公司可能都有其他的政策/規則。為了確保系統安全，我們必須確保我們的系統符合這些策略並處於合規狀態。

傳統上，合規檢查是可追溯的且大部需要手工完成的過程。這絕對是個錯誤的方法。合規性是我們必須始終確保的事情，因為這是限制任何潛在威脅因素或關閉任何可能威脅因素的唯一方法。

處理配置管理自動化（也稱為期望狀態配置-DSC）的工具至關重要。它將確保我們的配置/設置始終配置在基線或所需的狀態上。

監視和滲透測試對於驗證和確保此合規性至關重要。Nessus，Nmap 或 metasploit 等工具可用於測試系統的安全性。在測試過程中，監視和檢測系統應檢測到它們並發出警報。

用戶

快速一覽

- 教育，教育，教育
- 養成良好的習慣（例如鎖定電腦）

在任何系統中，使用者一般來說都是最薄弱的環節。為了確保使用者不易遭受網路釣魚攻擊或社交操縱，培訓和教育至關重要。我們必須確保使用者知道要查找的內容，如果他們不確定，可以升級到已知資源。

一種方法是類比網路釣魚攻擊，這樣使用者就可以保持審視的態度。我們還必須執行其他政策，例如電腦解鎖狀態下不要離開它或不要把密碼寫下來。

證書和認證

Nutanix 在堆疊的各個部分（內部和外部）都具有以下安全認證/資格：



- 通用準則 (Common Criteria)
 - 1996 年六國七方簽署了《資訊技術安全評估通用準則》≈ CC1.0。
1998 年美國、英國、加拿大、法國和德國共同簽署了書面認可協議。後來這一標準稱為 CC 標準，≈ CC2.0。CC2.0 版於 1999 年成為國際標準 ISO/IEC 15408。目前已經有 17 個國家簽署了互認協定，即一個 IT 產品在英國通過 CC 評估以後，那麼在美國就不需要再進行評估了，反之亦然。
 - *This is currently under re-certification as of March 2020
- 安全技術實現指南 (Security Technical Implementation Guides (STIGs))
 - DOD IA 和啟用 IA 的設備/系統的配置標準。自 1998 年以來，DISA 現場安全運營 (FSO) 通過提供《安全技術實施指南》，在提高 DoD (國防部) 安全系統的安全狀況方面發揮了關鍵作用。STIG 包含用於“鎖定”可能容易受到惡意電腦攻擊的資訊系統/軟體的技術指南。
- 聯邦資訊處理標準 FIPS 140-2
 - FIPS Publication 140-2 是 NIST 所發佈的針對密碼模組的安全需求 (Security requirements for cryptographic modules)。FIPS 140-2，這些標準和方針由 NIST 發佈，並作為聯邦資訊處理標準 (FIPS) 在政府機構廣泛採用。目前該標準的最新版本發表於 2002 年 12 月 3 日，其提供了密碼模組評測、驗證和最終認證的基礎。
 - FIPS 140-2 standard is an information technology security accreditation program for cryptographic modules produced by private sector vendors who seek to have their products certified for use in government departments and regulated industries (such as financial and health-care institutions) that collect, store, transfer, share and disseminate sensitive but unclassified (SBU) information.
- NIST 800-53
- NIST 800-131a
- ISO 27001
- ISO 27017
- ISO 27018

自動安全配置管理 Security Configuration Management Automation (SCMA)

Nutanix 安全引擎現在可以讓客戶在特定時間點的安全基線檢查，演進到在整個部署的生命週期裡持續進行安全基準線監控/自動修復，確保集群中所有 CVM 和 AHV 主機滿足安全基線要求。檢查內容完全滿足所有 STIGs 的安全檢查專案，當發現不滿足要求時，不需要客戶介入就可以自動將配置設定為符合安全的設置。



執行 Ad-hoc SCMA

SCMA 將按照配置好的頻率執行（預設是每小時），當然也可以根據要求立即執行。可以通過登錄 CVM 後執行如下命令來運行 SCMA 工具：

```
# Run on a single CVM  
  
sudo salt-call state.highstate  
  
# Run on all CVMs  
allssh "sudo salt-call state.highstate"
```

Nutanix 命令列介面（The Nutanix Command Line Interface (NCLI)）允許客戶設置不同的配置去滿足嚴格的安全需求。

CVM 安全配置 CVM Security Settings

通過 NCLI 裡下面的命令可以在集群範圍內配置 SCMA 策略。下面列舉所有的命令和功能：

獲得 CVM 安全配置：

```
ncli cluster get-cvm-security-config
```

此命令輸出當前集群的配置，預設輸出如下內容：

```
Enable Aide : false  
  
Enable Core : false  
  
Enable High Strength P... : false  
  
Enable Banner : false  
  
Enable SNMPv3 Only : false  
  
Schedule : DAILY
```

設置 CVM 登錄條：



此命令可以設置當登錄任何一個 Nutanix CVM 時，是否開啟或關閉 Department of Defense (DoD) knowledge of consent 登錄條。

```
ncli cluster edit-cvm-security-params enable-banner=[yes|no] #Default:no
```

自訂登錄條

預設情況下， DoD knowledge of consent 登錄條是啟用的，通過下面的步驟可以自訂登錄條 (登錄任意一個 CVM 用 nutanix 帳號運行):

1. 備份當前的 Banner

```
sudo cp -a /srv/salt/security/KVM/sshd/DODbanner /srv/salt/security/KVM/sshd/DODbannerbak
```

2. 通過 vi 命令修改當前 Banner

```
sudo vi /srv/salt/security/KVM/sshd/DODbanner
```

3. 在所有其他 CVM 上重複以上修改 Banner 的步驟

4. 使用前面的命令啟用 Banner

設置 CVM 密碼長度

以下命令可以啟用或關閉長密碼策略
(minlen=15,difok=8,remember=24).

```
ncli cluster edit-cvm-security-params enable-high-strength-password=[yes|no]
```

```
#Default:no
```

設置高級入侵偵測環境 (Advanced Intrusion Detection Environment (AIDE))



以下命令可以開啟或關閉每週運行的 AIDE 服務

```
ncli cluster edit-cvm-security-params enable-aide=[yes|no] #Default:no
```

設置 SNMPv3 only

通過以下命令開啟或關閉 SNMPv3 only traps.

```
ncli cluster edit-cvm-security-params enable-snmpv3-only=[true|false]
```

```
#Default:false
```

設置 SCMA schedule

以下命令設置 SCMA 運行頻率：

```
ncli cluster edit-cvm-security-params schedule=[HOURLY|DAILY|WEEKLY|MONTHLY]
```

```
#Default:HOURLY
```

虛擬化安全設置

The following commands have been added to NCLI to support cluster-wide configuration of the SCMA policy. The list below gives all commands and functions:

獲得虛擬化的安全配置

```
ncli cluster get-hypervisor-security-config
```

此命令輸出當前集群的配置，預設輸出內容如下：

```
Enable Aide : false  
Enable Core : false  
Enable High Strength P... : false  
Enable Banner : false  
Schedule : DAILY
```

設置虛擬化登錄條



此命令可以設置當登錄任何一個 Nutanix 虛擬化時，是否開啟或關閉 Department of Defense (DoD) knowledge of consent 登錄條。

```
ncli cluster edit-hypervisor-security-params enable-banner=[yes|no]
```

```
#Default:no
```

設置虛擬化的密碼強度

以下命令可以啟用或關閉長密碼策略
(minlen=15,difok=8,remember=24).

```
ncli cluster edit-hypervisor-security-params enable-high-strength-password=[yes|no] #Default:no
```

設置 Advanced Intrusion Detection Environment (AIDE)

以下命令可以開啟或關閉每週運行的 AIDE 服務.

```
ncli cluster edit-hypervisor-security-params enable-aide=true=[yes|no]
```

```
#Default:no
```

設置 SCMA schedule

以下命令設置 SCMA 運行頻率：

```
ncli cluster edit-hypervisor-security-params schedule=[HOURLY|DAILY|WEEKLY|MONTHLY]
```

```
#Default:HOURLY
```

鎖定集群

集群鎖定功能是將 CVM 設置為只能通過金鑰登錄，而不能使用用戶名和密碼訪問的一種方式。

在 Prism 介面右上角的齒輪功能表裡可以找到集群鎖定功能的配置項：

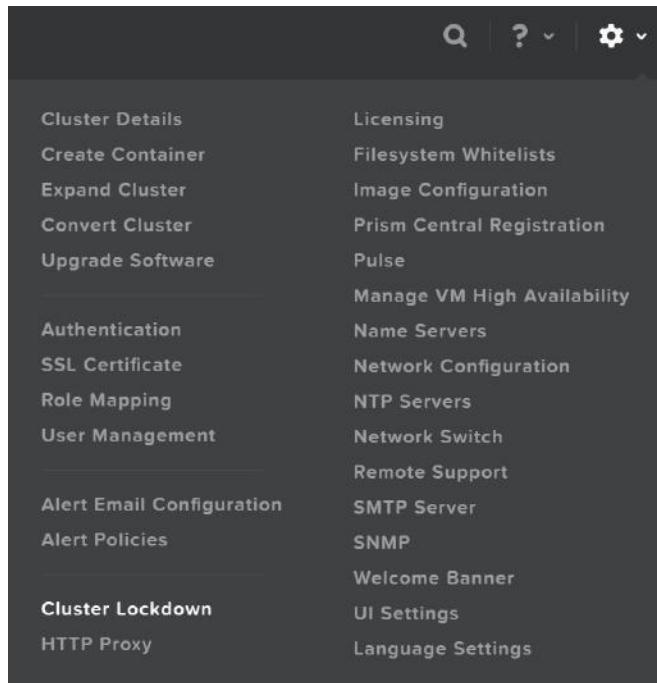


圖.集群鎖定菜單

這個介面會顯示當前的配置，並允許你添加/刪除用於訪問的 **SSH** 金鑰：

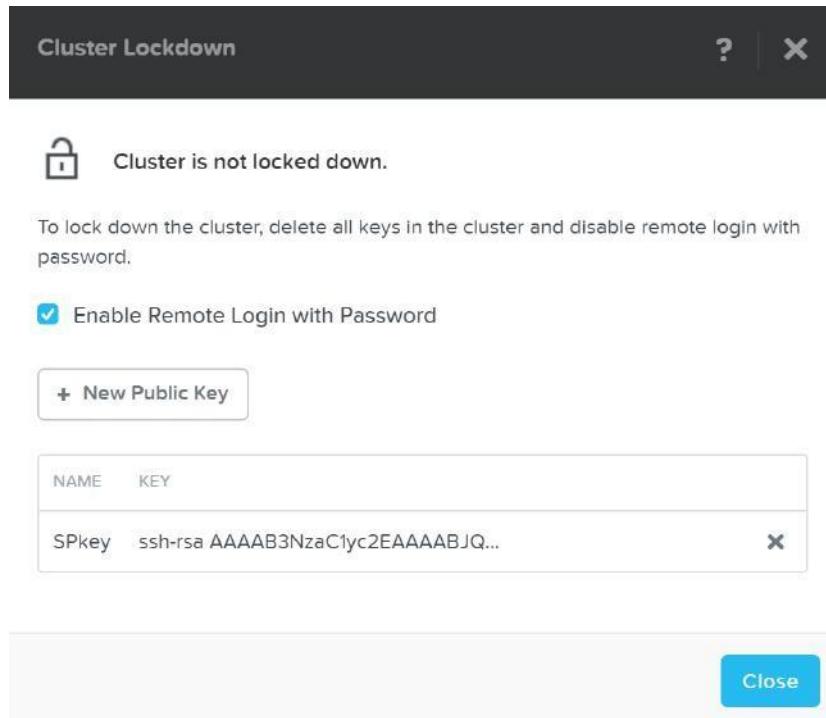


圖.集群鎖定介面

點擊“New Public Key”按鈕添加新的公開金鑰，並輸入金鑰資訊：

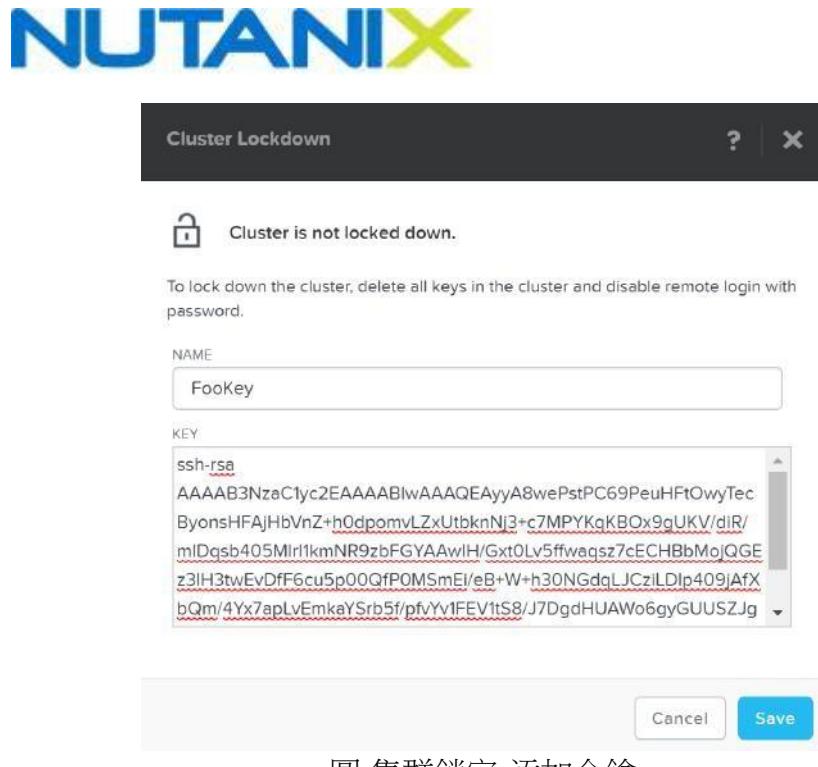


圖.集群鎖定-添加金鑰

SSH 金鑰的製作

可以運行下面的命令來生成一個 SSH 金鑰對：

```
ssh-keygen -t rsa -b 2048
```

這個命令會生成由兩個檔組成的金鑰對：

- `id_rsa` (私有金鑰)
- `id_rsa.pub` (公共金鑰 – 這個金鑰在向集群中添加金鑰時使用。)

在你添加了金鑰以後，你就有了有效的訪問憑證，你可以禁用基於用戶名密碼的登錄，通過勾選'Enable Remote Login with Password.'的選項，介面會彈出一個確認視窗，點擊“OK”按鈕之後，集群就被設置為鎖定了。

4.2.1 資料加密和金鑰管理

資料加密是一種允許對資料進行編碼的方法，只有那些被授權的人才能使用資料，使任何未經授權的人都無法解析。



例如，如果我有一條消息要發送給某人並確保只有他們可以讀取它，我可以使用密碼（金鑰）加密消息（明文）並向他們發送加密消息（密文）。如果此消息被盜或被攔截，則攻擊者只能看到密文，這些密文在沒有密碼來解密消息的情況下幾乎無用。一旦所需的一方收到了消息，他們就可以使用我們給他們的金鑰解密消息。

有幾種加密資料的主要方法：

• 對稱加密（私密金鑰加密）：

- 相同的金鑰用於加密和解密資料

- 示例：AES，PGP *，Blowfish，Twofish 等。

• 非對稱加密（公開金鑰加密）：

- 一個金鑰用於加密（公開金鑰），另一個金鑰用於解密（私

- 密金鑰）◦ 示例：RSA，PGP *等

注意：PGP（或 GPG）使用對稱和非對稱金鑰。

在討論資料加密時，通常在兩個主要環境中完成：

- 傳輸中：雙方之間傳輸的資料（例如，通過網路發送資料）

- 靜止：靜態資料（例如存儲在設備上的資料）

從 5.8 開始，Nutanix 解決了靜態資料加密問題。

以下部分將介紹 Nutanix 如何管理資料加密及其關鍵管理選項。

4.2.1.1 資料加密

Nutanix 通過三種主要可選途徑提供資料靜態加密：

- 原生基於軟體的加密 (FIPS-140-2 Level-1) *released in 5.5
- 使用自加密磁片 (SED) (FIPS-140-2 Level-2)
- 軟體 + 硬體加密

加密配置可選在集群或容器級別，這取決於虛擬化層的類型：

- 集群級別加密：
 - AHV, ESXi, Hyper-V
- 容器 級別加密：
 - ESXi, Hyper-V

注意：對於基於 SED 加密的部署，物理設備自身加密，所以是集群級別的。

通過在設置功能表（齒輪）中使用“Data-at-Rest Encryption”功能查看集群加密狀態。可以看到當前狀態並允許配置加密（如果當前沒啟用）。



下面例子中可以看到加密在集群層面已經啟用：



圖.資料加密-已啟用（集群層面）

下面例子中可以看到已經為列出的特定容器啟用加密：



Data-at-Rest Encryption

Encrypting your cluster will help keep your information safe.



[Manage Keys](#)

Encryption State of Cluster: Encrypt data by creating encrypted storage containers.

Encrypted Storage Containers

alert_test

alert_test2

alert_test3

圖.資料加密-已啟用（容器層面）

通過點擊“edit configuration”按鈕可以啟用/修改配置。彈出的功能表可以配置用於加密的 KMS 或者查看現在正在使用的 KMS：



All cluster data will be encrypted with software.

Select Key Management Server (KMS)

The KMS manages the encryption keys used to encrypt data.

Cluster's local KMS

Keep your keys safe with the cluster's local KMS. Prerequisites: The local KMS can only be used via software encryption and the cluster must contain at least three nodes.

An external KMS

Configure Key Management Servers and upload SVM certificates. You will manually download and upload certificates to validate the KMS. Nutanix recommends having two or more Key Management Servers for redundancy.

Save KMS Type

Back

圖.資料加密-配置

對於外部的 KMS，功能表可以引導使用 CSR 申請流程，配合 CA 完成簽名。

原生基於軟體的加密

Nutanix 軟體加密提供原生的 AES-256 靜態資料加密。這會和任何 KMIP 或者符合 TCG 規範的外部 KMS 伺服器 (Vormetric, SafeNet 等等) 或者 Nutanix 原生 KMS (5.8 引入，後面會介紹更多) 交互。對於加密/解密，系統會使用 Intel AES-NI 加速功能減少潛在的性能影響。

當資料寫入 (OpLog 和 Extent Store) 時，資料會在寫入磁片之前，在校驗值的邊界進行加密。

加密是資料寫入磁片之前應用到資料的最後一次轉化：

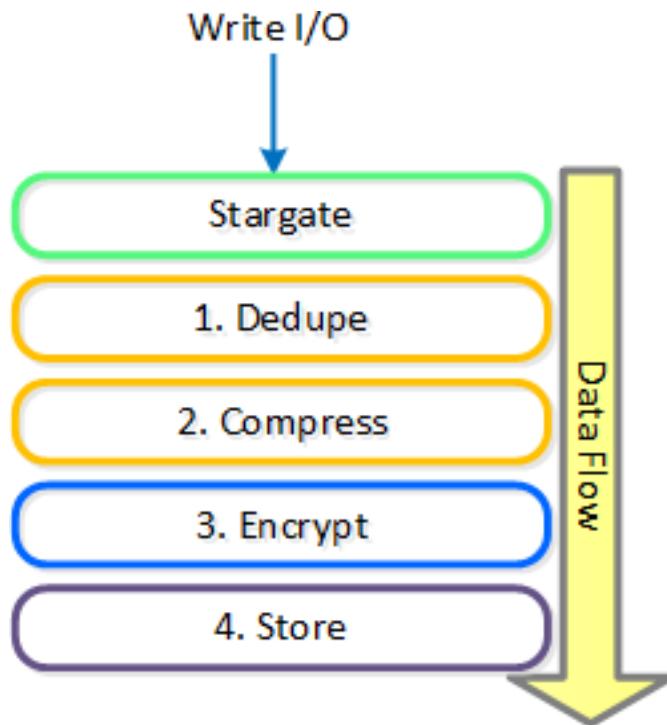


圖. 資料加密-轉化應用

加密和資料效率

由於資料加密是在應用所有去重壓縮之後，可以確保保持這些方法對空間的節省。簡單來說，對加密的資料的去重壓縮率會和非加密資料完全一樣。

讀數據時，會從磁片上校驗值邊界讀取加密資料，解密之後返回。在校驗值邊界執行加密/解密可以確保沒有讀放大產生。借助於 Intel AES-NI 卸載，可以看到幾乎沒有任何對性能/回應時間的影響。

基於 **SED** 的加密

下圖顯示了概要的架構圖：

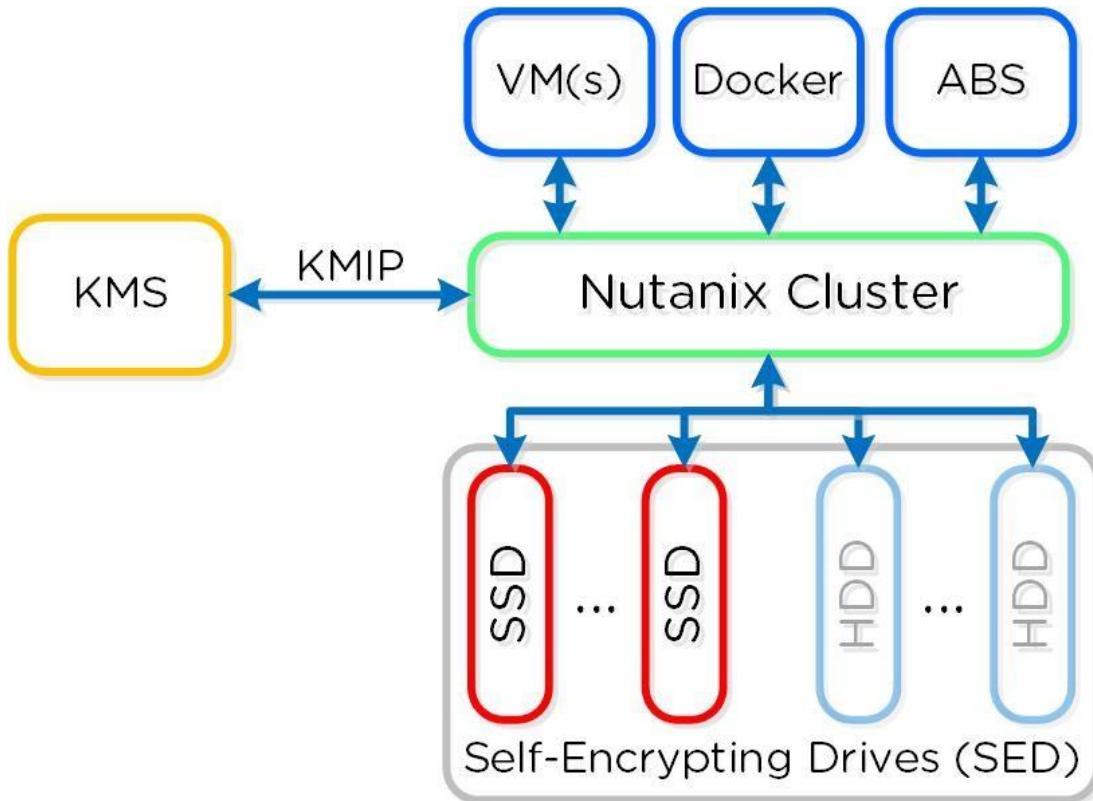


圖.資料加密-概要

SED 的資料加密方式是將存放裝置劃分為能設置為加密狀態和非加密狀態的“資料條帶”。在 Nutanix 的系統裡，boot 和 Nutanix 的 Home 分區是被加密處理過的。所有的資料磁片和資料條帶是用 big key 到 level-2 標準進行高度加密處理的。

當集群啟動的時候，它會向 KMS 伺服器發出請求解鎖磁片的金鑰。為了確保在集群裡從來也不緩存任何金鑰。在系統冷開機和 IPMI 重置的過程中，這個節點將需要向 KMS 伺服器發出獲取解鎖磁片的請求。CVM 軟啟動的過程不會強制發生以上操作。

4.2.1.2 金鑰管理(KMS)

Nutanix 提供原生的秘鑰管理（本地秘鑰管理器-LKM）存儲功能（5.8 引入）作為對協力廠商專用 KMS 解決方案的替代方案。用來消除對專用 KMS 的需要並簡化環境，當然外部 KMS 仍然支持。

前述章節提到過，秘鑰管理是所有資料加密方案最重要的部分。多個秘鑰用來在整個堆疊中提供非常安全的秘鑰管理方案。

在方案中使用了三種類型的秘鑰：

資料加密秘鑰 (DEK)

- 用來加密資料的秘鑰
- 秘鑰加密秘鑰 (KEK)

- 用來加密 DEK 的加密秘鑰

主加密秘鑰 (MEK)

- 用來加密 KEK 的加密秘鑰
- 只在使用本地秘鑰管理器時使用

下圖展示了不同秘鑰和 KMS 選項的關係：



圖. 資料加密-秘鑰管理



本地秘鑰管理（LKM）服務分佈在每個 Nutanix 節點，原生的運行在每個 CVM 中。服務使用 FIPS140-2 加密模型（通過認證），秘鑰管理器完成所有秘鑰管理動作（例如重新生成秘鑰，備份秘鑰等）的同時，對於最終用戶完全透明。

配置資料加密時，原生的 KMS 可以通過選擇“Cluster's local KMS”來使用：

All cluster data will be encrypted with software.

Select Key Management Server (KMS)

The KMS manages the encryption keys used to encrypt data.

Cluster's local KMS
Keep your keys safe with the cluster's local KMS. Prerequisites: The local KMS can only be used via software encryption and the cluster must contain at least three nodes.

An external KMS
Configure Key Management Servers and upload SVM certificates. You will manually download and upload certificates to validate the KMS. Nutanix recommends having two or more Key Management Servers for redundancy.

Save KMS Type

Back

圖. 資料加密-配置

主秘鑰使用 Shamir's Secret Sharing 演算法切片後存儲在集群的所有節點上保證彈性和安全。最小必須有 ROUNDUP(N/2)個節點重構秘鑰， N =集群中所有節點的數量。

秘鑰備份和秘鑰輪換

一旦啟用加密，推薦備份資料加密秘鑰（DEK）。如果備份，必須保證健壯的密碼和位置安全。

系統提供輪換（重新生成秘鑰）KEK 和 MEK 的功能。系統自動每年輪換主秘鑰，該操作也可以按需完成。在增加/移除節點事件中，也會發生主秘鑰輪換。



4.3 分散式存儲結構



分散式存儲 DSF 像集中存儲一樣呈現給 Hypervisor，然而所有的 I/O 是在本地處理以提供更高的性能。這些節點如何構成分散式系統的更詳細內容會在下面章節裡介紹。

4.3.1 資料結構

Acropolis 分散式存儲結構的高層元件如下：

存儲池

角色：一組物理存放裝置

描述：一個存儲池是一組物理存放裝置，包括集群內所有節點的 PCIe SSD、SSD 和 HDD。存儲池可以跨多個 Nutanix 節點並且隨著集群的擴展而擴展。在大部分情況下，建議單個集群配置一個存儲池。

容器

角色：一組虛擬機器或者檔的邏輯分組

描述：容器（container）從邏輯上劃分存儲池，並包含一組虛擬機器或者檔（即虛擬磁片）。一些配置項（例如 RF）是在容器層實現的，然後應用在單個虛擬機器文件層面。容器和 Datastore 是一一對應的（在 NFS、SMB 場景中）。

vDisk

角色：vDisk

描述：一個虛擬磁片檔(vDisk)是 DSF 上任意一個大於 512KB 的檔（包括 VMDK 和虛擬機器硬碟）。vDisk 邏輯上由組成‘Block map’的 vBlocks 組成，

最大 DSF vDisk 容量

DSF/stargate 方面並沒有強制限制 vdisk 的大小。4.6 版本，vdisk 尺寸數值佔用 64bit 二進位數字。這意味著理論上最大 vDisk 容量可以達到 $2^{63}-1$ or $9E18$ (9 Exabytes)。任意小於這個數值的限制都來自用戶端，比如 ESXi 規定的最大 vmdk 容量。

下圖顯示了各元件在 DSF 與 Hypervisor 之間的對應關係：

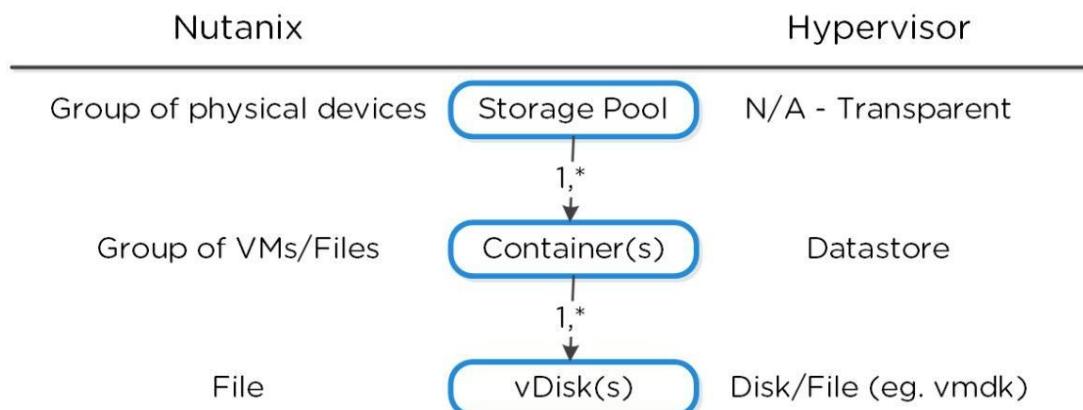


圖. 檔案系統分解縱覽

vBlock

- Key Role: 1MB chunk of vDisk address space
- Description: A vBlock is a 1MB chunk of virtual address space composing a vDisk. For example, a vDisk of 100MB will have 100 x 1MB vBlocks, vBlock 0 would be for 0-1MB, vBlock 1 would be from 1-2MB, and so forth. These vBlocks map to extents which are stored as files on disk as extent groups.

Extend

角色：邏輯上連續的資料塊

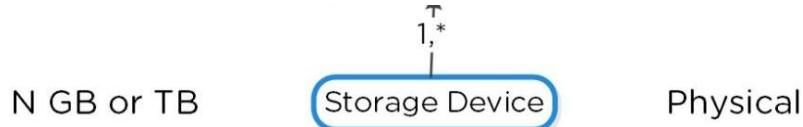
描述：一個 **Extent** 是邏輯上連續的 1MB 大小的資料塊，它由 n 個連續的 **block** 組成（**block** 大小取決於不同作業系統）。**Extent** 的讀寫修改是基於更小的子塊（也稱為 **slice**）以保證可細微性和有效性。當一個 **extent** 的 **slice** 被讀入 **cache** 時，可以根據被讀取/**cache** 的資料量大小被修剪。

Extend 組

角色：物理上連續的資料塊

描述：一個 **Extent Group** 是物理上連續存儲的 1MB 或者 4MB 大小的資料塊。它以檔的形式由 **CVM** 管理並保存在磁片上。**Extent** 被動態分佈在多個 **Extent** 組中，提供資料跨節點和磁片的條帶功能用來提高 IO 性能。注：在 **NOS4.0** 中，**Extent** 組可以是 1MB 或者 4MB，這取決於消重功能。

下圖說明這些不同結構分類和各個檔案系統之間的對應關係：



*Typically 4-8KB

圖.檔案系統分解細節

這是另外一個表明檔案系統各部分關係的示意圖：

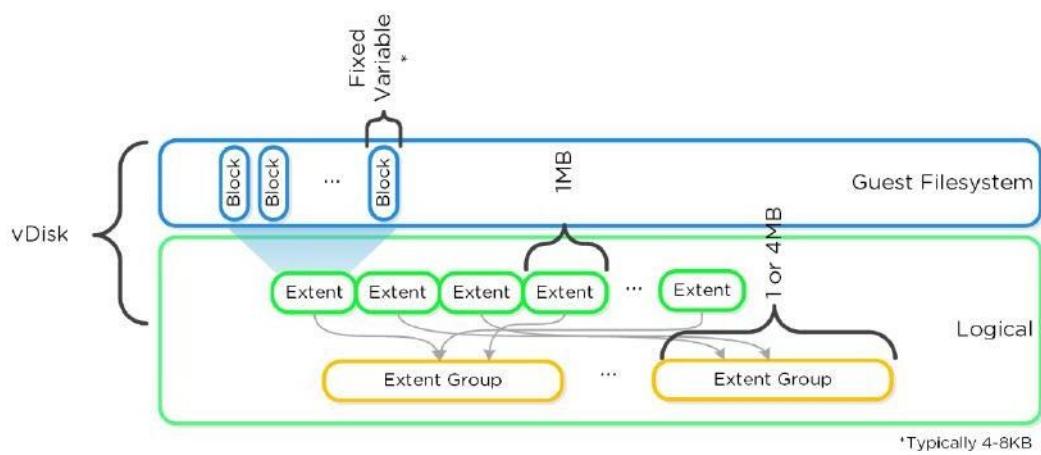


圖.圖形檔案系統分解

4.3.2 I/O 路徑和緩存

你可以通過觀看下面視頻來說明理解：<https://youtu.be/SULqVPVXefY>

典型的超融合存儲 I/O 路徑可分為以下重要的幾層：



1. Guest OS (UVM) 到虛擬磁片

- Nutanix 對此保持不變。根據 hypervisor，guest OS 將使用設備驅動程式與虛擬磁片設備進行對話。取決於 hypervisor，它可以是 `virtio-scsi` (AHV)，`pv-scsi` (ESXi) 等。虛擬磁片也將根據管理程式而有所不同（例如，`vmdk`，`vhd` 等）。

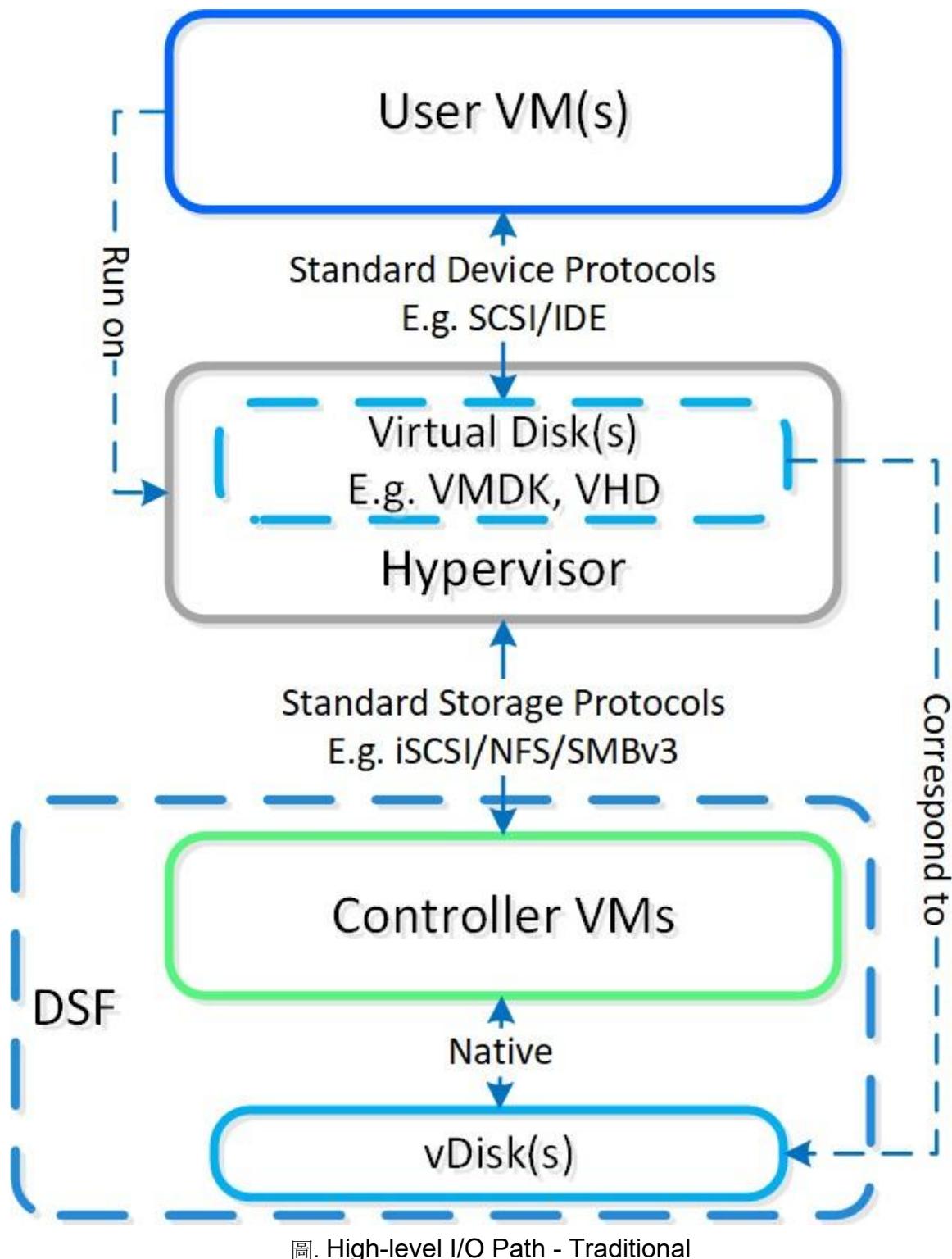
2. Hypervisor 到 Acropolis DSF (通過 CVM)

- Hypervisor 和 Nutanix 之間的通信是通過標準存儲協定（例如 iSCSI，NFS，SMBv3）在 CVM 和 hypervisor 的本地介面上進行的。此時，所有通信都在該主機本地執行（在某些情況下，I/O 將是遠端的（例如，本地 CVM 關機等）。

3. Nutanix I/O 路徑

- 對於 hypervisor 和 UVM 都是透明的，並且對於 Nutanix 平臺是本地的。

下圖顯示了這些層 high-level 的關係：



4.3.2.1 通信 和 I/O

在 CVM 中，Stargate 進程負責處理所有存儲 I/O 請求以及與其他 CVM/物理設備的交互。存放裝置控制器直接傳遞到 CVM，因此所有存儲 I/O 都繞過 hypervisor。

下圖顯示了傳統 I/O 的 high-level 路徑：

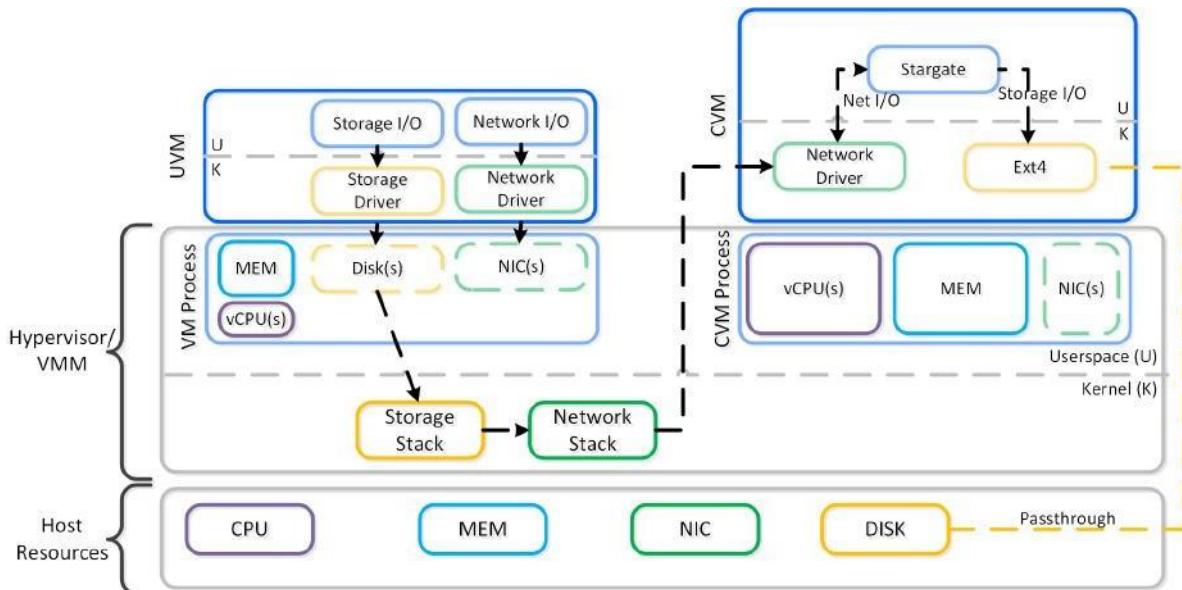


圖. High-level I/O Path

在 5.18 中，引入 Nutanix BlockStore，以提供一個可擴展的檔案系統層和塊管理層，所有這些都在用戶空間中處理。這就從設備中刪除了 Ext4，並消除了 Ext4 對內核驅動程式的調用。引入了新型的存儲介質（例如 NVMe），設備利使用者空間庫直接處理設備 I/O（例如 SPDK），而無需進行任何系統調用（上下文切換）。通過 BlockStore + SPDK 的組合，所有 Stargate 設備交互都已移入到用戶空間，從而消除了上下文交換或內核驅動程式的調用。

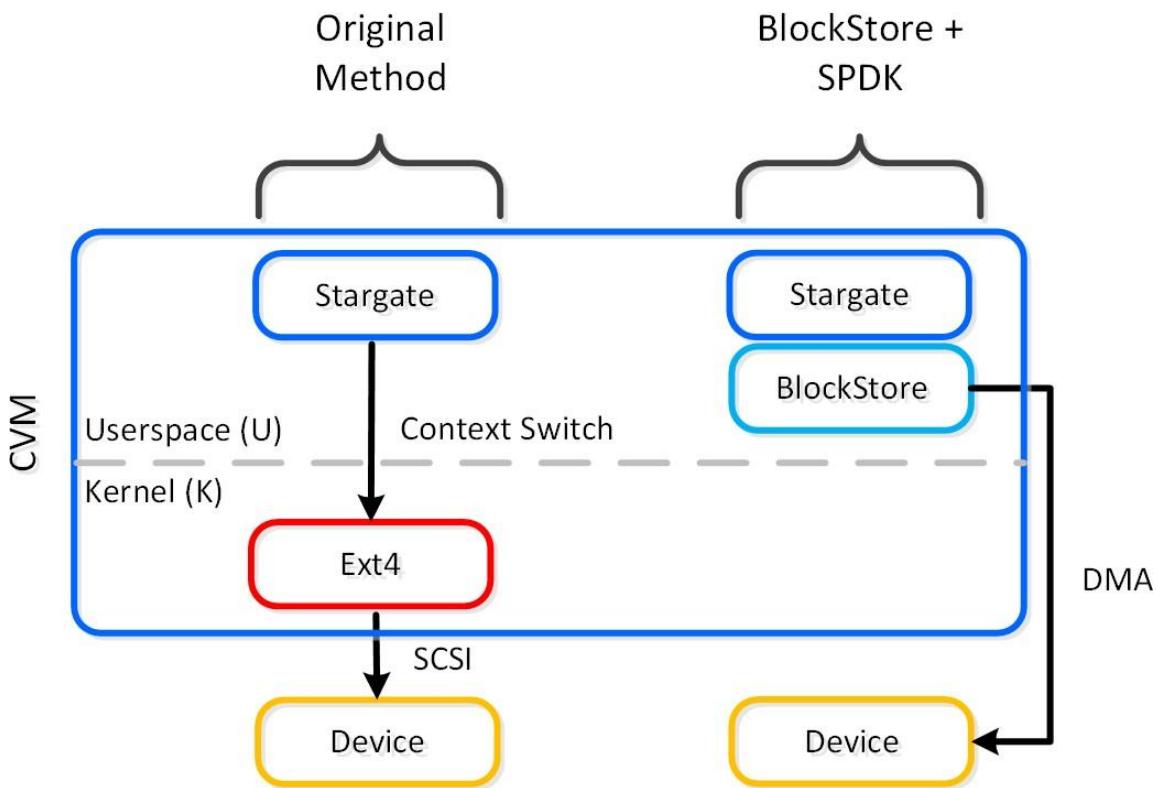


圖. Stargate - Device I/O Path

下圖顯示了使用 BlockStore + SPDK 更新的 I/O 路徑的高級概覽

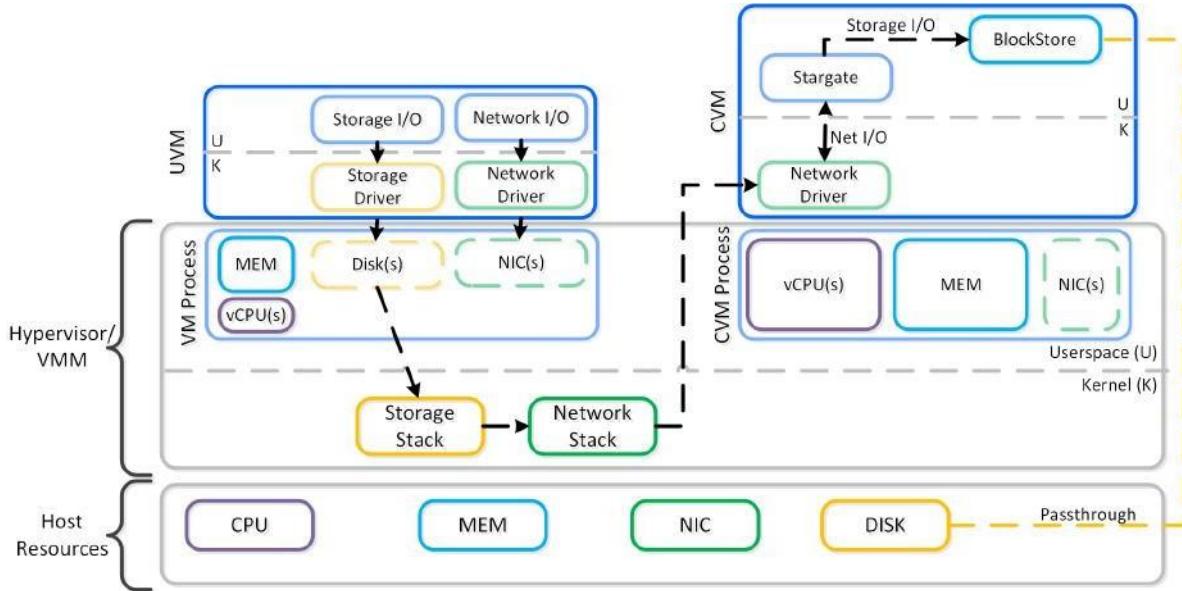


圖. High-level I/O Path - BlockStore

使用預設堆疊將調用內核級驅動程式來執行 CVM 跨網路通信進行資料複製。

但是，啟用 RDMA 這些 NIC 繞過虛擬機器管理程式中的任何內容傳遞到 CVM。同樣在 CVM 中，所有使用 RDMA 的網路流量僅使用內核級驅動程式作為控制路徑，然後所有實際資料 I/O 在使用者空間中完成，而無需任何上下文切換。

下圖顯示了 RDMA 的 I/O 路徑的高級概覽：

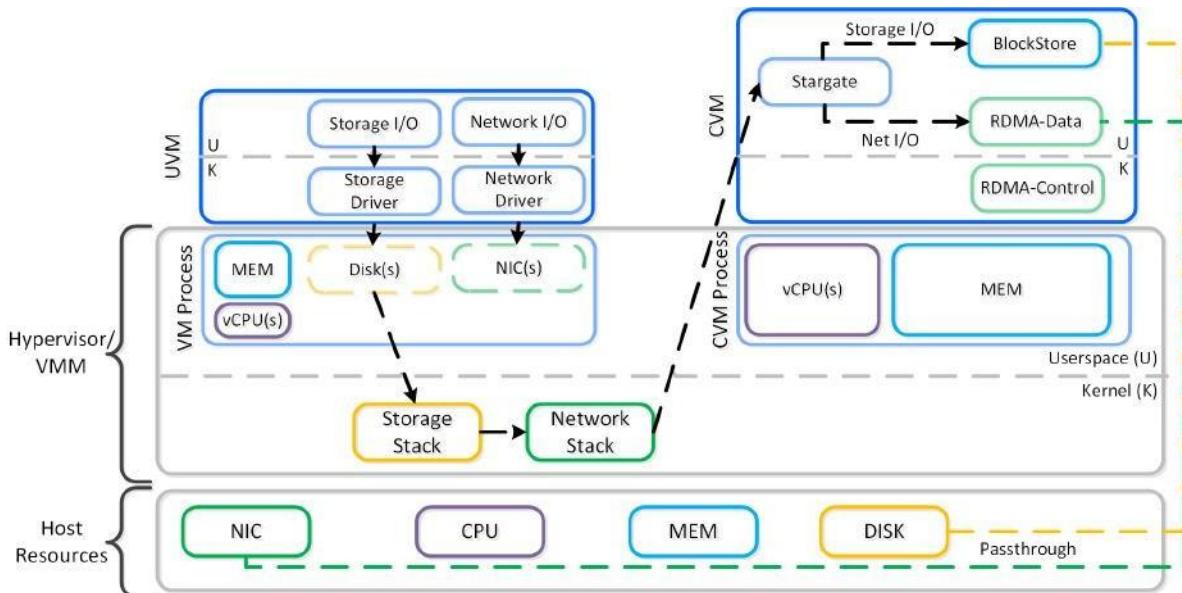


圖. High-level I/O Path - RDMA

歸納一下，以下增強功能在如下方面做了優化：

1. PCI 直通繞過管理程式進行設備 I/O
2. SPDK + Blockstore 消除了內核存儲驅動程式的交互，並將其移至用戶空間
3. RDMA 繞過管理程式，所有資料傳輸均在 CVM 用戶空間中完成

4.3.2.2 Stargate I/O 邏輯

在 CVM 中，Stargate 進程負責處理來自使用者 VM (UVM) 和持久性 (RF 等) 的所有 I/O。當向 Stargate 發出寫入請求時，有一個寫入特徵器，它將確定寫是否持久保存到 OpLog，Extent Store 或 AES 中。同樣對於讀取，讀取表徵器負責處理讀取和管理緩存/預讀。

Nutanix I/O 路徑由以下高級元件組成：

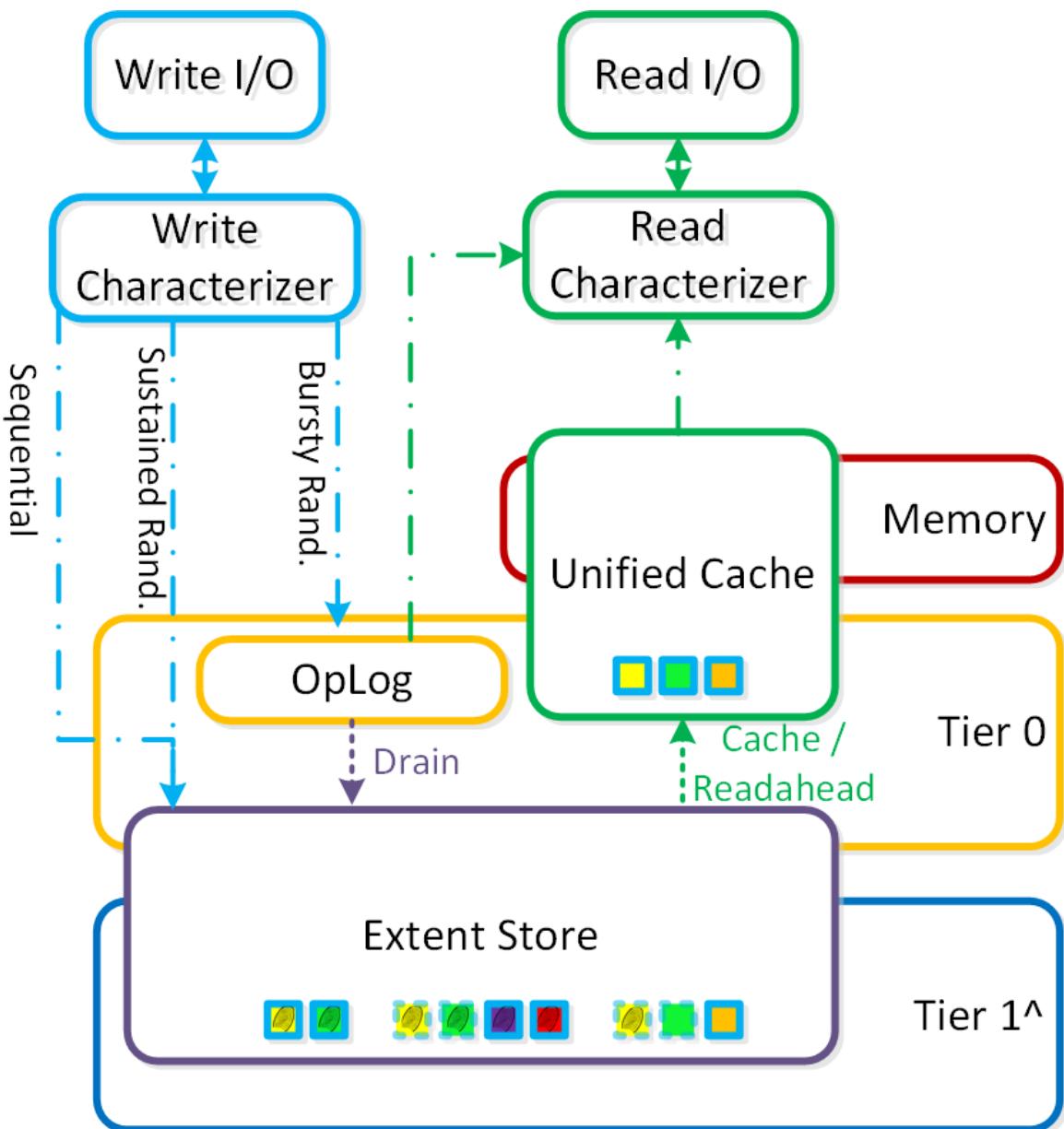


圖. DSF I/O Path

*從 AOS 5.10 開始，在滿足必要條件時可以使用自治區存儲（AES）來處理持續的隨機工作負載。

在全快閃記憶體節點配置中，擴展存儲將僅由 SSD 設備組成，不會發生層 ILM，因為僅存在一個快閃記憶體層。如果使用混合快閃記憶體（例如 NVMe，Intel Optane 等+ SATA SSD），則性能最高的介質將是 Tier 0，而性能較低的介質將是 Tier1。對於混合

（並非所有快閃記憶體）方案，快閃記憶體將是 第 0 層，硬碟為第 1 層。

4.3.2.3 OpLog



角色：持久性寫緩存

描述：**Oplog** 類似於檔案系統的日誌（**journal**），用來處理突發的寫操作，並聚合這些寫操作順序地寫到 **Extent Store** 中。為了保證資料高可用性的要求，在寫操作完成（**Ack**）之前，資料寫入 **Oplog** 的同時被同步複製到另一個 **CVM** 的 **Oplog** 中。所有 **CVM** 的 **Oplog** 都參與複製操作，並依據 **CVM** 負載情況自動選擇。

Oplog 保存在 **CVM** 的 **SSD** 層以提供極高的 **IO** 寫性能，特別是隨機 **IO** 寫操作。對於順序的 **IO** 寫操作會直接寫到 **Extent Store** 上而繞過 **Oplog**。如果資料當前在 **Oplog** 中，所有的讀請求會直接從 **Oplog** 中回饋，直到資料被合併推送到 **Extent Store** 中後，只有當讀的資料不在 **Oplog** 中，讀請求才會從 **Extent Store/Unified Cache** 中獲得。如果在容器中的指紋（消重功能）被啟用時，則所有寫 **IO** 操作時，資料塊會被標記指紋，以便在資料進入 **Unified Cache** 時進行消重操作。

4.3.2.3.1 Per-vDisk OpLog Sizing

vDisk 的 **Oplog** 計算

Oplog 是共用資源，但分配是在 **vDisk** 基礎上完成確保每個 **vDisk** 有均等機會。這由每個 **vDisk Oplog** 限制（**Oplog** 中每個 **vDisk** 的最大資料量）實現。帶有多個 **vDisk(s)** 的虛擬機器可以使用每個 **vDisk** 限制乘以磁片數量的 **Oplog**。

每個 **vDisk Oplog** 限制當前是 6GB（4.6 開始），以前版本最大 2GB。

由下面的 **Gflag** 控制：`vdisk_distributed_oplog_max_dirty_MB`。

4.3.2.4 Extent Store

角色：持久性資料存儲

描述：**Extent Store** 是 **DSF** 中持久性大型存放區組件，它橫跨 **SSD** 和 **HDD**，並且能擴展到其他節點的存放裝置上。有兩種資料流程進入 **Extent Store**，一種是從 **Oplog** 中被合併的資料順序寫入到 **Extent Store**，另外一中是繞過 **Oplog** 的順序寫操作直接寫入 **Extent Store** 中。**Nutanix** 的 **ILM** (**information lifecycle management**) 基於 **IO** 類型 (**IO Pattern**) 動態決定資料保存的熱分層位置，並且在不同熱分層之間移動數據。

順序寫特徵

當寫到 **vDisk** 的併發寫 **IO** 超過 1.5MB 時（4.6 開始），這個寫 **IO** 就被認定為順序寫。符合條件的 **IO** 會旁路 **Oplog** 直接到 **Extent Store**，因為大塊對齊資料不會受益於 **IO** 合併。

由下面的 **Gflag** 控制：

`vdisk_distributed_oplog_skip_min_outstanding_write_bytes`。

所有其他 **IO**，包括較大的（例如，超過 64K）仍然由 **Oplog** 處理。



4.3.2.5 Autonomous Extent Store (AES)

- 關鍵角色：持久資料存儲
- 說明：自治擴展存儲區（AES）是一種在 AOS 5.10 中引入的在擴展存儲區中寫入/存儲資料的新方法。它利用了主要是本地+全域中繼資料的混合（在下面的“可伸縮中繼資料”部分中有更多詳細資訊），從而由於中繼資料的局部性而實現了更加有效的持續性能。對於持續的隨機寫入工作負載，這些負載將繞過 OpLog 並使用 AES 直接寫入擴展存儲區。對於突發的隨機工作負載，這些將採用典型的 OpLog I/O 路徑，然後在可能的情況下使用 AES 排入擴展存儲。注意：從 5.11.1 開始，要啟用 AES，該節點必須至少具有 8 個快閃記憶體設備，或者如果至少一個設備是 NVMe，則該節點必須具有任意數量的快閃記憶體設備。

4.3.2.6 Unified Cache

角色：動態讀緩存

描述:Unified Cache 是可消重的讀緩存，它橫跨 CVM 的記憶體和 SSD。當讀取的資料不在緩存中（或基於一個特定的指紋），資料會放到 Unified Cache 的 Single-touch 池，該池完全保留在記憶體中，並且使用 LRU(最近最少使用演算法)直到資料被彈出。如果後續有對該資料的讀取操作，資料會被“移動”（其實沒有實際的資料移動，移動的只是資料的中繼資料（metadata）到 Multi-touch 池（Multi-touch 池跨記憶體和 SSD）中的記憶體段。這時開始，資料塊具備 2 個 LRU，一個是其位於記憶體段的 LRU，如果 LRU 耗盡，則資料被移動到 Multi-touch 池的 SSD 段，並被賦予一個新的 LRU。如果資料再次被讀取，則資料被移動到 Multi-touch 池的頂端，並重置其位於記憶體段的 LRU。指紋標記可以在 Containter 層面通過管理 UI 進行配置。預設指紋標記被禁用。

下圖是統一緩存中邏輯描述圖：

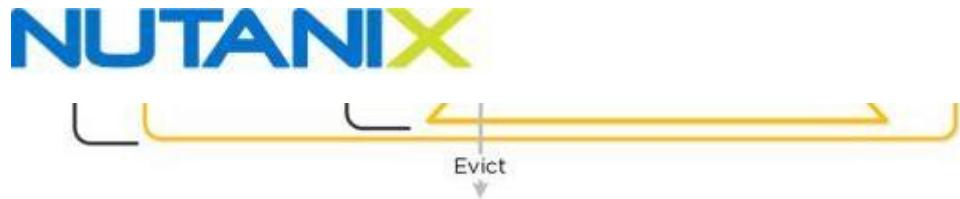


圖. DSF 統一緩存

緩存的細微性和邏輯：

資料是以 4K 細微性讀進緩存的，所有緩存是即時完成的，沒有延遲，也沒有採用批次處理方式把資料讀進緩存中。

每個 CVM 有它自己的本地緩存來管理本地 VM 的 vDisk，當一個 vDisk 被克隆的時候（例如：新的克隆，快照等）每個新的 vDisk 都有自己對應的塊，原有 vDisk 標記為不改變。這可以確保每個 CVM 可以有自己的基於原 vDisk 的緩存副本來確保快取一致性。

當發生覆蓋寫的時候，會被重定向到 VM 自己的對應塊中新的 extent。這確保了不會發生任何緩存崩潰。

Extent Cache

角色： 記憶體中的讀緩存

描述：Extent Cache 是完全位於 CVM 記憶體中的讀緩存。當指紋標記和消重被禁用時，它用來存儲沒有被標記指紋的 Extents。在 3.5 版本中 Extent Cache 和 contentCache 是相互獨立的，但在 4.5 版本這兩個 Cache 被合併成統一的 Cache。

4.3.3 可擴展的中繼資料

你可以通過觀看下面視頻來說明理解：<https://youtu.be/MIQczJhQI3U>

中繼資料（Metadata）是任何智慧系統的核心，對於檔案系統和存儲陣列來說是至關重要的。在 DSF 中，有一些關鍵的結構來保證 DSF 擴展到超大規模資料量時依然可靠。



- 必須 100%時間裡都是正確的（稱為“強一致性”）

- 必須符合 ACID 策略
- 必須具有無限擴展性
- 必須沒有任何擴展性上的瓶頸（必須是線性擴展）

從 AOS 5.10 開始，中繼資料分為兩個區域：全域中繼資料與本地中繼資料（之前所有中繼資料都是全域的）。這樣做是對“中繼資料當地語系化”進行優化，並限制系統用於中繼資料查找的網路流量。

這樣更改的基礎是因為並非所有資料都需要是全域的。例如，每個 CVM 不需要知道一個特定 extent 位於哪個物理磁片上，他們只需要知道哪個節點保存該資料，而僅那個節點需要知道哪個磁片具有該資料。

通過這樣做，我們可以限制系統存儲的中繼資料的數量（為僅本地資料消除中繼資料 RF），並針對“中繼資料當地語系化”進行優化。

下圖顯示了全域中繼資料與本地中繼資料之間的區別：

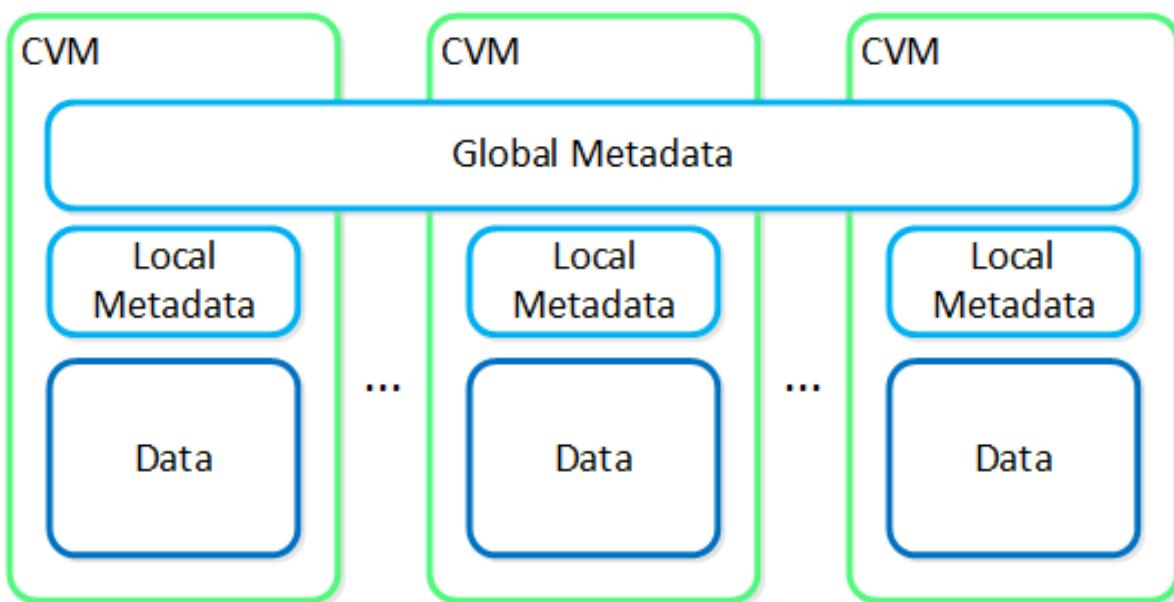


圖. Global vs. Local Metadata

4.3.3.1 本地中繼資料

- 說明：
 - 每個 CVM 的本地中繼資料存儲，其中僅包含本地節點所需的資訊。5.10 中引入的自治區存儲（AES）充分利用了這一點。
- 存儲機制：



- AES DB (基於 Rocksdb)
- 存儲的資料類型：
 - 物理 extent / extent group 放置 (例如，egroup 到磁片的映射) 等

4.3.3.2 全域中繼資料

- 說明：
 - 中繼資料可全域供任何 CVM 使用，並在群集中的 CVM 之間分片。 5.10 之前的所有中繼資料。
- 存儲機制：
Medusa Store (基於 Cassandra)
- 存儲的資料類型：
 - vDisk 塊映射，extent 到節點的映射，時間序列狀態，配置等。

以下部分介紹了如何管理全域中繼資料：

在上述架構章節中提到，DSF 使用一種“環狀”的 Key-Value 結構的分散式資料庫來保存重要的中繼資料和其它平臺資料 (例如 stats 等)。為了確保中繼資料的可用性和冗餘度，也同樣引入了複製因數 (RF)。一旦一條中繼資料(Metadata)被寫或者更新後，這條記錄將同時寫到“環”中的另一個節點，然後被複製到 n 個其他節點 (n 決定與集群的大小)。集群中大多數 (majority) 節點同意之前確保所有資訊一致，這就是強一致性的 Paxos 演算法。這確保了 Nutanix 平臺資料的“強一致性”以及中繼資料作為平臺的一部分進行存儲。

下圖顯示了在一個 4 節點集群中，中繼資料的插入和更新操作：

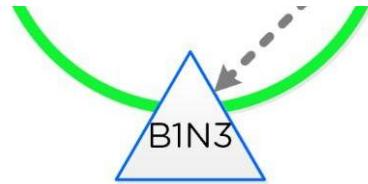


圖.Cassandra 環形結構

對於 DSF 的中繼資料資料庫，集群擴展時的性能也是至關重要的。與傳統的“雙控”和主備模式不同，每個 Nutanix 節點只負責整個集群中繼資料中的一部分。這種方式消除了傳統的性能瓶頸問題，並且允許中繼資料被集群中所有節點共同維護。並且使用“一致性散列演算法（Consistent Hashing）”來保證當節點數量變化時，需要被 remapping 的中繼資料量最少。當節點數量從 4 增加到 8 個時，新節點將被插入到環中各個老節點之間，使用類似的 block 感知能力提供可靠性。

下圖顯示了環被擴展時的情況：

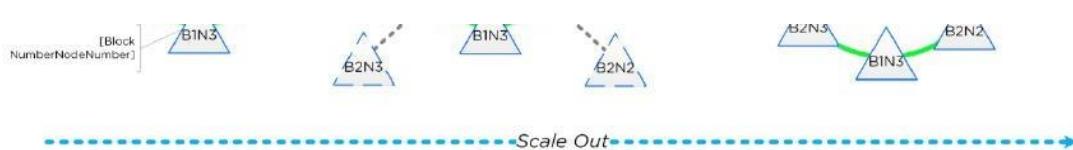


圖.Cassandra 橫向擴展

4.3.4 資料保護

你可以通過觀看下面視頻來說明理解：<https://youtu.be/OWhdo81yTpk>

Nutanix 平臺使用複製因數 (RF - Replication Factor/Resillience Factor) 和校驗和 (Checksum) 來保證當節點或者磁片失效時，資料的冗餘度和可用性。按照上面的解釋，Oplog 作為暫存區來接受所有寫操作，並保存到低延遲的 SSD 層上。當資料寫入本地 Oplog 時，資料被“同步”複製到另 1 個或者 2 個 Nutanix CVM 的 Oplog 之中（取決於 RF 設置），當這個操作完成之後，此次寫操作才被確認 (Ack)。這樣能確保資料至少存在於 2 個或者 3 個獨立的節點上，保證資料的冗餘度。注：當 RF 為 3 時，至少需要 5 個節點，並且中繼資料 (metadata) 資料會保留 5 份 (RF5)。

為每部分資料 (1GB 的虛擬磁片資料) 選擇 OpLog 對等體，並且所有節點都積極參與。選擇對等體的多個因素（例如回應時間，業務，容量利用等）。這消除了零碎並確保每個 CVM / OpLog 可以同時使用。

使用者資料的 RF 是通過 Prism 在容器層面進行配置。所有節點都參與 Oplog 的複製操作，這樣能消除“熱點節點”，並保證線性的性能擴展。當資料被寫入時，同時計算該資料塊的校驗和，並且作為資料塊中繼資料中的一部分進行存儲。隨後資料塊在保證滿足 RF 的前提下，被“非同步”推送到 Extent Store 中。當發生節點或者磁片失效，資料塊會重新在所有節點間進行複製以滿足複製因數的設置。任何時候，讀取資料塊並同時計算其校驗和以確保資料塊有效。當資料塊檢查結果不匹配校驗和時，副本資料將會覆蓋該無效資料塊。

即使未發生活躍 I/O，資料也始終被監視以確保完整性。Stargate 的擦洗器操作將持續掃描擴展區組，並在磁片未被大量使用時執行校驗和驗證。這可以防止比特腐爛或損毀的磁區。

下圖顯示了這一過程的邏輯圖：

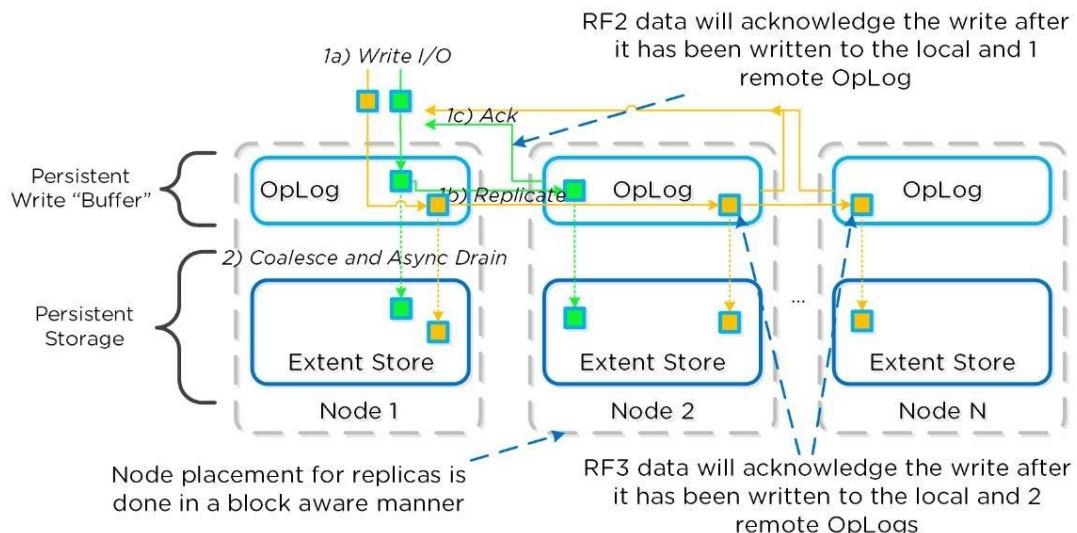


圖. DSF 資料保護

4.3.5 可用域 (主機殼感知) 知)



你可以通過觀看下面視頻來說明理解：<https://youtu.be/LDaNY9AJDn8>



可用域（也稱為節點/主機殼/機架感知）是分散式系統的關鍵結構，以遵守元件和資料放置的決定。DSF 現在是節點和主機殼感知的，然而，隨著集群規模的增長，需要提升到機架感知。Nutanix 說的 **block** 是指一個主機殼，包含一個，兩個或者四個伺服器節點。注：如果需要實現主機殼感知，最少需要 3 個主機殼，否則缺省是節點感知。

Nutanix 目前支援以下級別的感知模式：

- 磁片(所有版本)
- 節點(所有版本)
- 硬體設備(AOS 4.5 以上)
- 機櫃(AOS 5.9 以上)

建議使用統一的較新的主機殼來保證主機殼感知是允許的。通常的場景和使用的感知級別能夠在本章末找到。需要 3 個主機殼是用於確保符合規定要求(quorum)。例如，3450 是一個擁有 4 個節點的主機殼。把角色和資料分散到不同主機殼的原因是確保當一個主機殼故障或需要維護時，系統可以不中斷地持續運行。注：在一個主機殼裡，冗餘的電源和風扇是僅有的共用元件。

注意:機架感知要求管理員定義設備硬體設備所在的“機架”。

下圖現實了如何在 Prism 裡進行配置：

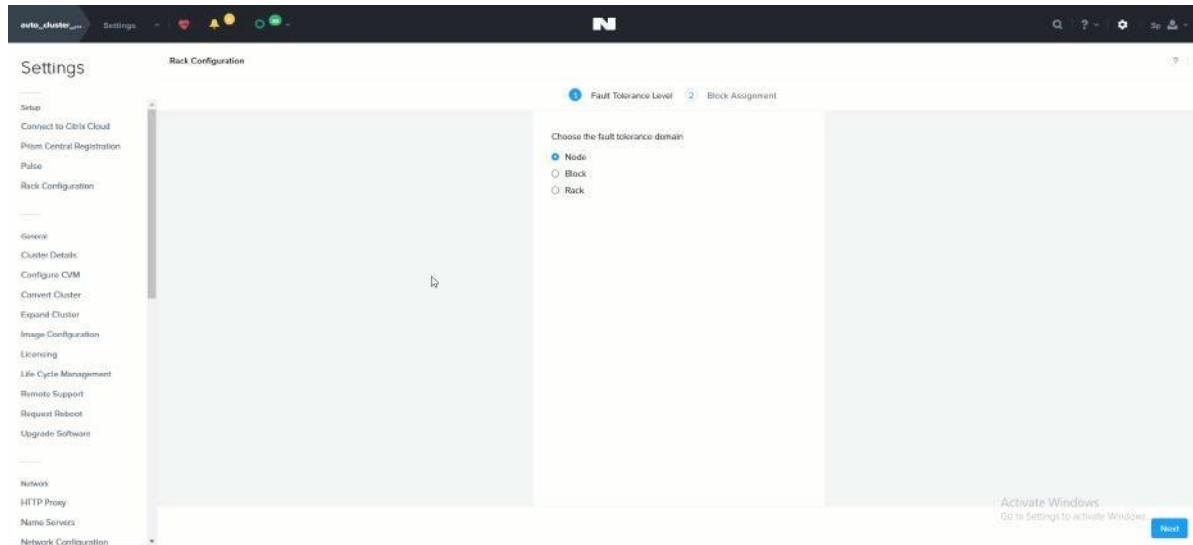


圖. 機櫃配置

機櫃感知可以分為以下幾個關鍵領域：

- 資料（虛擬機器資料）
- Metadata (Cassandra)



- 配置資料 (Zookeeper)

數據

DSF 的資料副本將會被寫到集群中的其它主機殼以保證當一個主機殼故障或者計畫停機時，資料仍然可用。這適用於 RF2 和 RF3，以及在主機殼故障的場景。一個簡單的比較是節點感知，當一個節點故障時，副本將被複製到另外的節點以提供保護。主機殼感知進一步增強了，當主機殼故障時提供資料的可用性保證。

下圖展示了在一個 3 主機殼的部署中副本如何放置的例子：

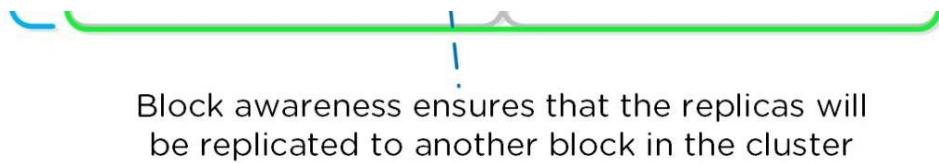
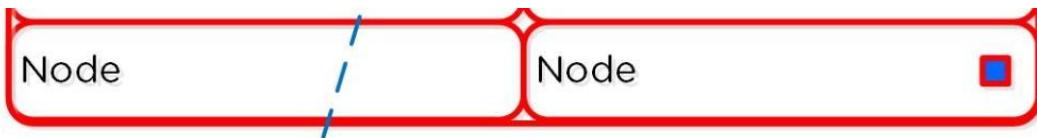


圖. 主機殼感知的副本放置

當主機殼故障時，主機殼感知將被維持，副本將重新複製到其他主機殼中：



In the event where full block awareness cannot be fulfilled node awareness will still be fulfilled to maintain the RF

圖. 主機殼故障情況下的副本放置

機架/硬體設備感知與 **Metro** 集群

一個常見的問題是，您能否跨兩個位置(房間、建築物等)跨集群，並使用硬體設備/機櫃感知來提供位置故障的彈性。

雖然在理論上這是可能的，但這並不是推薦的方法。讓我們先思考一下我們想要實現什麼：

1. 低 RPO
2. 低 RTO (HA 事件而不是 DR 事件)

如果我們以第一種情況為例，即我們試圖實現 $RPO \sim 0$ ，那麼最好利用同步或准同步複製。這在獲得相同的 RPOs 的同時，風險更低。

為了最小化 RTO，可以在同步複製的基礎上利用 **Metro** 集群，並將任何故障作為 HA 事件處理，而不是執行 DR 恢復。

總體來說，利用同步複製/城域群集，是出於以下原因：

- 通過同步複製/城域集群可以達到相同的最終結果，避免任何風險並保持故障域的隔離。
- 在不受支援的“延伸”部署中，如果兩個網站之間的網路連接斷開，一側將停止運行，這是因為必須維持仲裁（例如，主導方將保持運行）。在城域集群應用場景中，兩邊網站可以繼續獨立運行。
- 在故障場景中，資料的可用性域放置是盡力而為的
- 兩個網站之間額外的延遲/的網路頻寬減少會影響“延伸”部署群集的性能

感知條件和容錯

下面我們劃分為一些通用的場景以及哪個級別的容錯將被使用：

動態感知類型	FT 級別	是否打開糾刪	最小單元	容忍同時失效
節點	1	否	3 節點	1 節點
節點	1	是	4 節點	1 節點
節點	2	否	5 節點	2 節點
節點	2	是	6 節點	2 節點
主機殼	1	否	3 主機殼	1 主機殼
主機殼	1	是	4 主機殼	1 主機殼
主機殼	2	否	5 主機殼	2 主機殼
主機殼	2	是	6 主機殼	2 主機殼
機櫃	1	否	3 機櫃	1 機櫃
機櫃	1	是	4 機櫃	1 機櫃
機櫃	2	否	5 機櫃	2 機櫃
機櫃	2	是	6 機櫃	2 機櫃

作為 Acropolis 基礎軟體版本 4.5 及以後，主機殼感知是盡力而為的，不會強制啟用。這樣可以確保存儲資源不平衡（例如，重存儲節點）不會關閉該功能。當然，使用統一一致的主機殼最小化存儲不平衡仍然是最佳實踐。



4.5 以前，下列必須滿足下列條件：

如果 SSD 或者 HDD 層在不同主機殼間的差別>最大方差：節點感知

- 如果 SSD 或者 HDD 層在不同主機殼間的差別<最大方差：主機殼+節點感知

最大方差計算： $100/(RF+1)$ ，例如 RF2 是 33%，RF3 是 25%

中繼資料

正如在可擴展的中繼資料一章提到的，Nutanix 利用一個深度修改過的 Cassandra 平臺來存儲中繼資料和其它重要的資訊。Cassandra 利用一個環形架構，複製到環裡 n 個對等（peer）來保證資料的一致性和可用性。

下圖顯示了一個 12 節點集群的 Cassandra 環的例子：



圖. Cassandra 節點環

Cassandra 對等複製在整個環裡用順時鐘的方式反覆運算到各節點。在主機殼感知下，對等（peer）會分佈到主機殼裡，確保沒有兩個對等會在同一個主機殼裡。

下圖顯示了一個節點佈局按照上述環形轉換到主機殼佈局的例子：



For example any update on node 1 would be replicated to the next n nodes which would all exist on different blocks

圖. Cassandra 節點塊感知放置

在主機殼感知特性下，即使發生主機殼故障的情況下，仍然有至少兩份資料副本（中繼資料 RF3，在大型集群可以利用 RF5）。

下圖顯示了所有節點複寫拓撲形成環的例子

The following shows all of the connections between peers and the replication topology in a block aware model

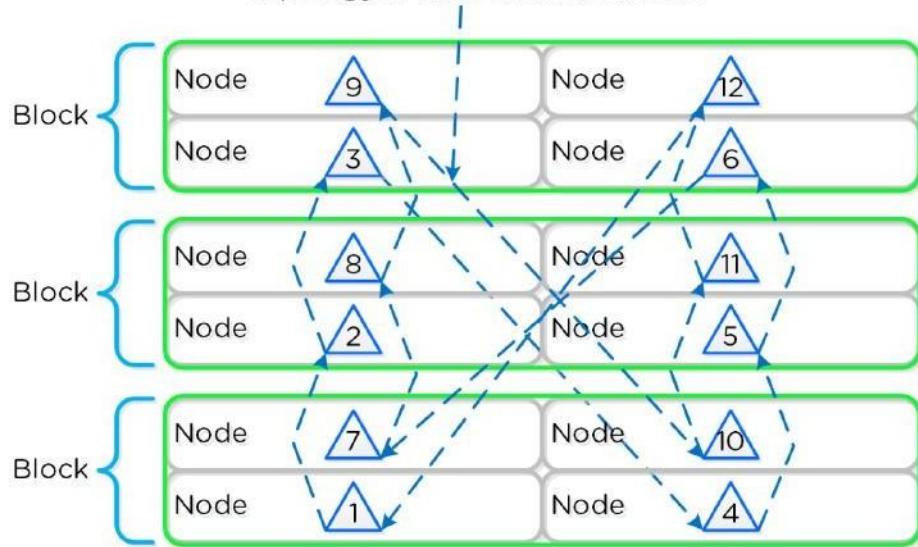


圖. 全 Cassandra 節點塊感知放置

下圖顯示了 3 個 Zookeeper 節點分佈在一個主機殼感知的方式的例子：

配置資料

中繼資料感知條件

下面我們看一下一些常見的場景和那個級別的感知會被使用：

- **FT1** (資料 **RF2**/中繼資料 **RF3**) 將會是主機殼感知，如果：

- **>3** 主機殼
- **X** 代表主機殼最大節點數，剩餘主機殼的節點數最少有 **2X** 個

例 1：4 個主機殼分別有 2, 3, 4, 2 個節點，不會主機殼感知，因為 $2+3+2 < 2*4$

最大節點的主機殼有 4 個節點，其他三個主機殼需要有 2×4 (8) 個節點。如果剩下的只有 7 個節點的情況下就不會觸發主機殼感知。

例 2：4 個主機殼分別有 3, 3, 4, 3 個節點，主機殼感知，因為 $3+3+3 > 2*4$

最大節點的主機殼有 4 個節點，其他三個主機殼需要有 2×4 (8) 個節點。如果剩下 9 個節點的情況下會觸發主機殼感知。

- **FT2** (數據 **RF3**/中繼資料 **RF5**) 將會主機殼感知

- **>5** 主機殼
- **X** 代表主機殼最大節點數，剩餘主機殼的節點數最少有 **4X** 個

例 1:6 個主機殼分別有 2, 3, 4, 2, 3, 3 個節點，不會主機殼感知，因為 $2+3+2+3+3 < 4*4$

最大節點的主機殼有 4 個節點，其他三個主機殼需要有 4×4 (16) 個節點。如果剩下的只有 13 個節點的情況下就不會觸發主機殼感知。

例 2:6 個主機殼分別有 2, 4, 4, 4, 4, 4 個節點，主機殼感知，因為 $2+4+4+4+4+4 > 4*4$

最大節點的主機殼有 4 個節點，其他三個主機殼需要有 4×4 (16) 個節點。如果剩下 18 個節點的情況下會觸發主機殼感知。



Nutanix 利用 Zookeeper 來存儲集群裡必要的配置資料。這些角色也使用主機殼感知的方式進行分佈以確保主機殼故障時仍然可用。

下圖顯示了一個啟用主機殼感知時 3 個 Zookeeper 節點分佈情況的例子：

圖 . Zookeeper 塊感知放置

在一個主機殼失效的事件裡，意味著其中一個 Zookeeper 節點沒有了，ZooKeeper 的角色將會被轉移到集群裡的另外一個節點上，如下圖示：

圖. Zookeeper 塊放置失敗

當這個主機殼恢復正常上線後，Zookeeper 角色將會被轉移回來，以維持主機殼感知。注：4.5 版本前，遷移需要手工方式，不能自動進行。

4.3.6 資料路徑冗餘

你可以通過觀看下面視頻來說明理解：https://youtu.be/SJlb_mTdMPg

可靠性和彈性是 DSF 或者任何主要存儲平臺設計的關鍵。

傳統的架構是基於硬體應該是可靠的思路建設的，相反，Nutanix 使用一個不同的方法，它認為硬體始終會出故障的。基於此，系統設計使用簡潔和不中斷的方式來處理這些故障。

注：但這並不意味著 Nutanix 的硬體品質不過關，只是概念的轉變。Nutanix 硬體和品質部門一直致力於高品質和審核過程。

在之前的章節提到過，中繼資料和資料使用集群故障冗餘級別的複製因數進行保護。5.0 支持 FT1 和 FT2 的故障冗餘等級，分別對應中繼資料 RF3 和資料 RF2，或者中繼資料 RF5 和資料 RF3。

要瞭解更多關於中繼資料如何切片的內容請參考之前的“可擴展中繼資料”章節。
要瞭解更多關於如何保護資料的內容請參考之前的“資料保護”章節。

在通常狀態中，集群資料擺放類似於下圖：

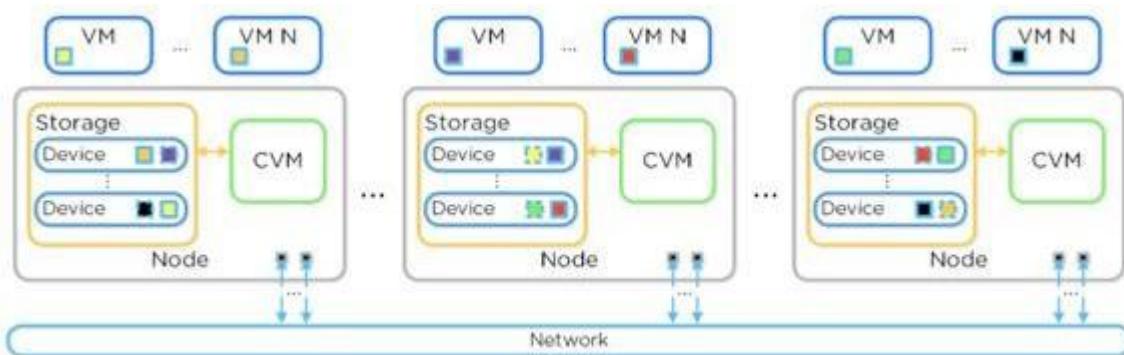


圖. 資料連結冗餘- Normal State

正如看到的，虛擬機器 vDisk 資料在磁片上有 2 份或者 3 份拷貝，分佈在節點之間和相關聯的存放裝置之間。

資料分佈的重要性

依靠保證中繼資料和資料跨所有節點和所有磁片設備分佈，可以在通常的資料處理和重新保護時保證最高性能。

當資料進入系統，主拷貝和副本拷貝會分佈在本地和所有其他遠端節點。這樣就消除了潛在熱點（例如運行緩慢的節點和磁片），並確保一致的寫性能。在資料磁片或者節點故障時資料必須被重新保護，集群的全部能力都可以被用來重建。在重建事件裡，中繼資料掃描（找出故障設備上的資料和副本的位置）平均分佈在所有 CVM 上。一旦資料副本被所有健康 CVM 發現，磁片設備

（SSD + HDD）和主機網路上聯鏈路可以被用來並行重建資料。

例如，在一個有磁片故障的 4 節點集群中，每個 CVM 會處理 25% 的中繼資料掃描和資料重建。在一個 10 節點集群中，每個 CVM 處理 10% 的中繼資料掃描和資料重建。在一個 50 節點集群中，每個 CVM 會處理 2% 的中繼資料掃描和資料重建。

關鍵點：使用 Nutanix，確保資料的一致性分佈可以保證一致的寫性能和極其優秀的重新保護時間。這也會應用到任何集群範圍的活動（例如糾刪碼，壓縮，重刪等）。

對比其他解決方案，使用 HA 對或者單個磁片保存所有資料的完整拷貝，如果鏡像的節點／磁片遭受壓力（面對重 IO 或者資源受限）會面臨前端性能問題。

而且，在資料必須被重新保護的故障事件中，這些方案會受限於單個控制器，單個節點的磁片資源和單個節點網路上聯鏈路。當 TB 級數據必須被重新保護，這會嚴重受限於本地節點的磁片和網路頻寬，增加系統停留在潛在的由於其他故障發生導致資料丟失狀態的時間。

潛在故障的級別



作為分散式系統，DSF 是基於處理元件、服務和 CVM 故障而構建的，可以分為如下幾個級別：

- 磁片故障
- CVM 故障
- 節點故障

什麼時候重建開始？

當出現計畫外故障時（在某些情況下，如果它們無法正常工作，我們將主動將其置於離線狀態），我們立即開始重建過程。

與其他一些等待 60 分鐘開始重建並且在此期間僅保留一份副本的供應商不同（風險很高，如果出現任何類型的故障，可能會導致資料丟失），我們不願意犧牲這種風險。更高的存儲利用率

我們可以這樣做是因為 **a**) 我們的中繼資料的細微性 **b**) 動態地選擇用於寫入 RF 的對等體（當出現故障時，所有新資料（例如新寫入/覆蓋）保持其配置的冗餘）和 **c**) 我們可以處理事物 在重建期間重新連線並在資料經過驗證後重新接收資料。在這種情況下，資料可能會“過度複製”，其中 Curator 掃描將啟動並刪除過度複製的副本。

磁片故障

磁片故障的特點是磁片被拔掉、磁片徹底壞了或者由於 I/O 出錯被主動地刪除。當 Stargate 看見 I/O 錯誤或者經過一段時間磁片沒有回應，該磁片將被標記下線。一旦發生上述情況，Hades 將運行 S.M.A.R.T. 並檢查設備狀態。如果測試通過磁片會被標為線上，如果失敗會保持離線狀態。如果 Stargate 多次標記一個磁片下線（每小時 3 次），即使 S.M.A.R.T. 通過測試，Hades 仍會停止再次標記磁片線上。

虛擬機器影響：

- HA 事件：沒有
- I/O 失敗：沒有
- 延遲：沒有影響

當出現磁片故障時，會立即觸發一個 Curator 掃描（MapReduce 框架）。將會掃描中繼資料（Cassandra）以發現故障磁片上的資料和哪些節點/磁片擁有副本。

一旦發現資料需要重新複製，它會把複製任務分派到整個集群的節點。

當有壞盤或 SMART 日誌監控到錯誤的時候，磁片自查 Drive Self Test (DST)的進程被啟動。

下圖顯示了磁片故障和重新保護的例子：

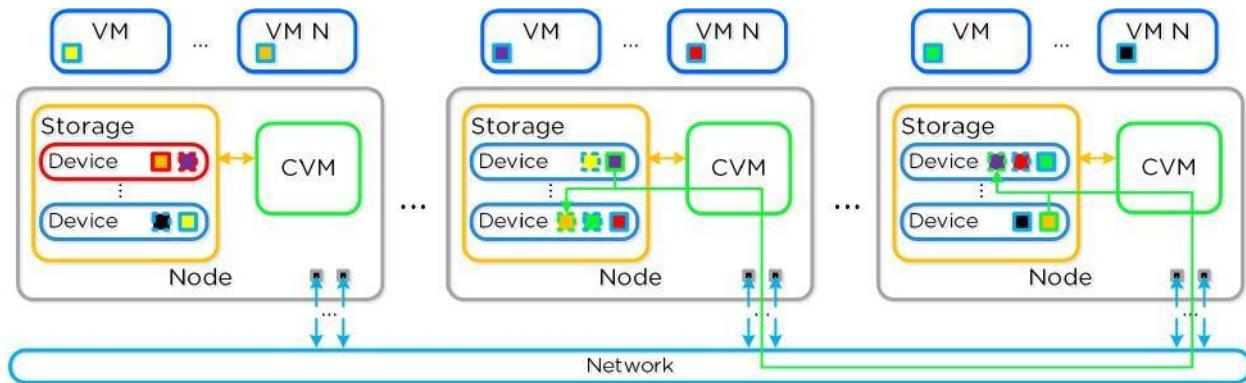


圖. 資料連結冗餘 – 磁片故障

一個重要的事情需要在這裡強調的是鑑於 Nutanix 把資料和副本分散到所有節點 /CVMs/磁片，所有節點/CVMs/磁片將參與到重新複製中。

這樣實質上減少了重新回到保護狀態的時間，因為可以利用整個集群的能力。集群越大，越快回到保護狀態。

節點故障

虛擬機器影響：

- HA 事件：發生
- I/O 失敗：沒有
- 延遲：沒有影響

如果一個節點失效了，虛擬機器 HA 機制將啟動，將會在另外的節點上重新啟動虛擬機器。一旦被重新啟動，虛擬機器將在本地 CVM 接管下繼續提供服務。類似於磁片失效，Curator 掃描將發現失效的資料節點和冗餘的資料節點。與磁片失效類似，節點失效也會在所有正常的集群節點內實現被保護資料的再平衡。

當節點故障持續相當長的時間，失效的 CVM 將被從中繼資料環中刪除，當節點恢復和穩定一段時間後，CVM 將被重新加回環中。

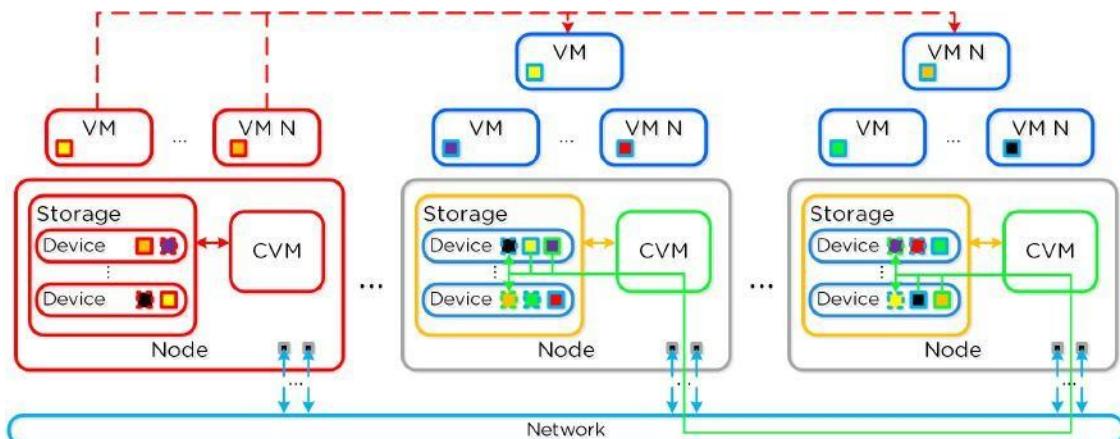


圖. 資料連結冗餘- 節點故障

如果延長一段時間後（在 4.6 版本是 30 分鐘）節點仍然故障，故障的 CVM 將會被從中繼資料環上移除。它恢復並穩定一段時間後會被重新加入環。

專家提示

資料可靠性狀態會在 Prism 的主頁面顯示

您也可以通過 cli 查詢資料冗餘性狀態：

```
# Node status
ncli cluster get-domain-fault-tolerance-status type=node

# Block status
ncli cluster get-domain-fault-tolerance
```

這將是最新更新過的資料，但是您可以發起 Curator 半掃描重新刷新這個資料。

CVM 故障

CVM 故障的特點是由於 CVM 的權利行為引起 CVM 臨時不可用。系統設計本身可以很好的透明地處理這些故障。當 CVM 故障時，I/O 將被重定向到集群的其它 CVM。具體機制因 Hypervisor 不同而各異。CVM 的輪流升級過程實際上是利用了這個能力，每次升級一個 CVM，反覆運算到整個集群。

虛擬機器影響：

- HA 事件：沒有
- I/O 失敗：沒有



- 延遲：由於 I/O 經過網路，潛在的高延遲

在出現 CVM 故障的時候，由原來故障 CVM 處理的 I/O 將會被轉發到集群裡的其它 CVM 上。ESXi 和 Hyper-V 是通過一個叫 CVM 自動路徑的進程來處理的，利用 HA.py(比如"happy")，修改路由，把發送到內部位址 (192.168.5.2) 的資料包轉發到集群裡其它 CVM 的外部位址。這樣使得資料存儲保持原封不動，只是 I/O 由遠端 CVM 來處理。

一旦本地 CVM 恢復正常並且穩定了，路由將被刪除，本地 CVM 也將接管新的 I/O。

對於 KVM，是利用 iSCSI 多路徑技術，主路徑是本地 CVM，另外兩個路徑是遠端。當主路徑故障時，其中一個遠端路徑將被啟動。

類似於 ESXi 和 Hyper-V 的自動路徑技術，當本地 CVM 恢復正常後，本地 CVM 將接管成主要路徑。

4.3.7 容量優化

Nutanix 平臺集成了多種存儲優化技術並協同工作，使任何工作負載的可用容量被有效利用，這些技術具備智慧和自我調整的工作特性，消除了需要手動配置和微調。

下面是一些集成的優化技術：

- 累積碼
- 壓縮
- 消重

有關如何在以下各節中找到這些功能的更多詳細資訊。

該表描述了哪些優化適用於高級工作負載：

資料轉換	應用	說明
累積碼	所有	在不影響正常寫入或讀 IO 性能的同時提供比傳統的 RF 更高的可用性，且降低開銷。在磁片/節點/塊故障的情況下，同時資料必須被解碼的情況下有一些讀的開銷
線上壓縮	所有	不影響隨機 IO，有助於提高存儲層利用率。通過減少資料從磁片複製和讀取，有利於大的 IO 或順序的 IO 性能
離線壓縮	無	線上壓縮將僅僅壓縮大的或連續的寫入，而隨機或小的 IO/應該採用離線壓縮處理

Perf Tier 消重	P2V/V2V,Hyper-V (ODX),Cross-container clones	為沒有克隆或通過採用 Acropolis 克隆的資料提供更大的緩存效率
容量層消重	P2V/V2V,Hyper-V (ODX),Cross-container clones	擁有上面的好處同時，減少在磁片上的開銷

4.3.7.1 紅刪碼

Nutanix 平臺依賴於 **RF** 來保護資料和可用性，這個方法提供了最高程度的可用性，因為當資料失效時，不需要從其他存儲位置或者重新計算來讀取。然而，由於需要完全複製，存儲資源的成本比較高。

為了提供一個可用性和降低存儲容量的平衡，**DSF** 提供了使用紅刪碼（**EC**）來對資料進行編碼。

和 **RAID**（級別 4, 5, 6 等）的概念相似，校驗位是計算出來的。**EC** 對不同節點上的資料塊條帶進行編碼和計算校驗位元。當主機或者磁片故障時，校驗位元可以被利用為計算任何缺失的資料塊（解碼）。在 **DSF** 裡，資料塊是一個 **extend** 組，每個資料塊必須在不同的節點上，屬於不同的 **vDisk**。

對於冷讀取的資料，我們將更傾向將來自同一 **vDisk** 的資料塊分佈在各個節點上，以形成資料條帶（相同 **vDisk** 資料條）。這簡化了 **garbage collection (GC)**，因為在刪除 **vDisk** 的情況下可以刪除整個條帶。對於讀取的熱資料，我們更傾向將 **vDisk** 資料塊保留在節點本地，並使用來自不同 **vDisk** 的資料（跨 **vDisk** 條）組成該資料條。由於本地 **vDisk** 的資料塊可以是本地的，其他 **VM / vDisk** 可以組成條帶中的其他資料塊，因此可以最大限度地減少遠端讀取。如果讀取的冷資料條變熱，系統將嘗試重新計算該資料條並當地語系化資料塊。

條帶中資料和同位塊的數量可根據所需的容忍故障進行配置。該配置通常稱為<資料塊數> / <同位塊數>。

例如，**RF2** 級別的可用性 (**N+1**) 在一個條帶裡能夠包括 3 個或者 4 個資料塊和一個校驗資料塊（例如 3/1 或者 4/1）。**RF3** 級別的可用性 (**N+2**) 在一個條帶裡能夠包括 3 個或者 4 個資料塊和兩個校驗資料塊（例如 3/2 或者 4/2）。

EC + 主機殼感知

從 **5.8** 開始，**EC** 可以以塊感知方式放置資料和同位塊（在 **5.8** 之前，這是在節點級別完成的）。

升級到 **5.8** 後，現有的 **EC Containers** 不會立即更改為阻止阻止的放置。如果集群中有足夠的塊（條帶大小 $(k + n) + 1$ ）可用，則這些以前的節點感知帶將移動到塊感知。新的 **EC Containers** 將構建塊識別 **EC** 條。

預期的消耗可以用校驗塊數量/資料塊數量來計算。例如，一個 **4/1** 的條帶有 25% 的消耗，或者是 **1.25*** (**RF2** 是 **2***)。一個 **4/2** 條帶有 50% 的消耗或者是 **1.5*** (**RF3** 是 **3***)。

下表是列出了常見條帶大小和消耗的例子：

	FT1 (RF2 equiv.)		FT2 (RF3 equiv.)	
Cluster Size (nodes)	EC Strip Size (data/parity blocks)	EC Overhead (vs. 2X of RF2)	EC Strip Size (data/parity)	EC Overhead (vs. 3X of RF3)
4	2/1	1.5X	N/A	N/A
5	3/1	1.33X	N/A	N/A
6	4/1	1.25X	2/2	2X
7	4/1	1.25X	3/2	1.6X
8+	4/1	1.25X	4/2	1.5X

專家提示：

強烈建議集群裡節點的數量要比條帶大小組合（資料塊數量+校驗塊數量）至少加一，以便當節點故障時可以重建條帶。這樣避免了條帶重建時（由 Curator 自動進行）產生的計算超載。例如，一個 4/1 條帶應該在集群裡至少有 6 個節點。上面的表格是按照最佳實踐得出來的。

編碼是事後進行的，利用編碼是在後處理過程中完成的，並利用策展人 MapReduce 框架進行任務分發。由於這是一個後處理框架，傳統的寫 I/O 路徑不受影響。

一個正常使用 RF 的環境看起來像這樣的：





圖.典型 DSF RF 資料結構

在這個場景裡，我們有混合 RF2 和 RF3 資料的集群，主資料塊在本地，副本資料塊在集群的其它節點上。

當 Curator 運行完全掃描時，它會發現合適的 extend 組可以被編碼的。合適的 extend 組必須是“寫冷的”，也就是它們寫入超過一個小時以上。在合適的 extend 組被發現後，編碼的任務將通過 Chronos 分發和控制。

下圖展示了一個 4/1 和 3/2 條帶的例子：



圖.DSF 編碼條帶化- 保存前

一旦資料被成功地編碼（條帶和校驗計算後），extend 組的副本將被刪除。

下圖展示了運行 EC 節省存儲空間後的環境：

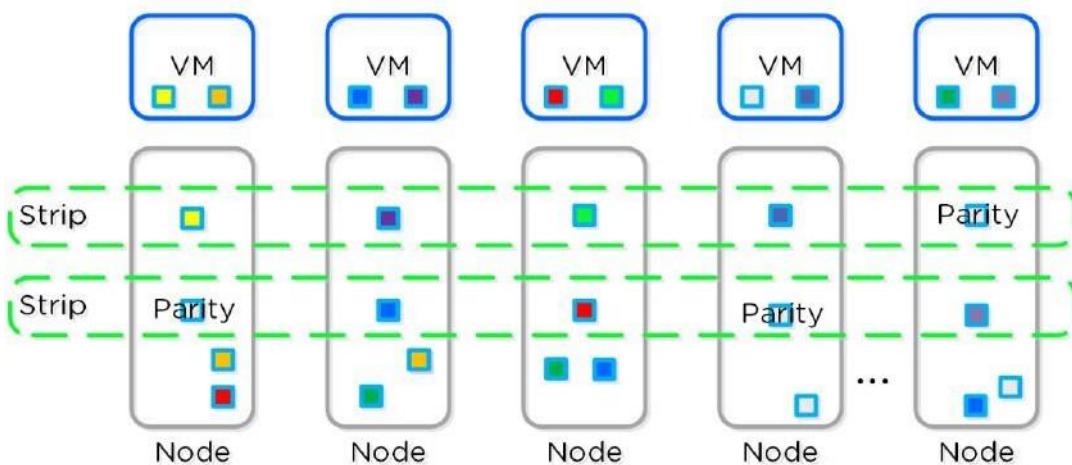


圖. DSF 編碼條帶化- 保存後

專家提示：糾刪碼和線上壓縮可以完美地結合使用，大大節省存儲空間。
在我們的環境裡，總是利用壓縮+EC 配置。

4.3.7.2 壓縮

你可以通過觀看下面視頻來說明理解：<https://youtu.be/ERDqOCzDcQY>

Nutanix 容量優化引擎（COE）負責資料的轉換來增加磁片中資料的效率。目前壓縮是 COE 實現資料優化的一項關鍵特性。DSF 提供線上和離線兩種壓縮方式以滿足客戶的需求和不同資料類型。5.1 後，離線壓縮被默認開啟。

當寫資料到 Extent Store (SSD+HDD) 時，線上壓縮會壓縮順序資料流程或者大 I/O (大於 64K)。包含從 Oolog 落下來的資料和不使用 Oolog 的順序資料。

Oolog 壓縮

截至 5.0 版本，Oolog 現在壓縮所有進入的大於 4K 的寫，並表現出良好的壓縮效果 (Gflag: vdisk_distributed_oolog_enable_compression)。這樣會更有效的利用 Oolog 空間並提升實際性能。

當資料從 Oolog 落入 Extent Store 會被解壓縮、對齊並被按照 32K 對齊的尺寸重新壓縮。(截至 5.1 版本)

本特性缺省開啟，不需要使用者配置。

離線壓縮開始按照正常方式寫資料（以不壓縮狀態），然後借助 Curator 框架在集群範圍壓縮資料。當線上壓縮開啟但是 I/O 都是亂數據，資料會以未壓縮的狀態寫入 Oolog，歸併，然後在記憶體中壓縮後寫入 Extent Store。



Nutanix 利用 LZ4 和 LZ4HC 通過 AOS 5.0 及更高版本進行資料壓縮。在 AOS 5.0 之前的版本中，穀歌使用 Snappy 壓縮庫，該庫可提供良好的壓縮率，最小的計算開銷和極快的壓縮/解壓縮速率。正常資料將使用 LZ4 壓縮，這在壓縮和性能之間提供了很好的融合。對於冷資料，將利用 LZ4HC 提供更高的壓縮比。

以下兩個類型被歸類為冷資料：

正常資料：

- 三天沒有讀寫訪問 (Gflag: curator_medium_compress mutable_data_delay_secs)

不可更改資料（快照）

- 一天沒有訪問 (Gflag: curator_medium_compress immutable_data_delay_secs)

下面圖示展現了一個線上壓縮和 DSF 寫 I/O 通道如何交互的例子：



圖.線上壓縮 I/O 路徑

專家提示：建議總是使用線上壓縮（壓縮延遲=0），它只壓縮大的/順序寫，不影響隨機寫的性能。

這也將增加 SSD 層的可用容量，增加有效性能並允許更多資料放置在 SSD 層。而且，對於寫入並線上壓縮的大或者順序資料，用於 RF 的複製也會傳輸壓縮過的資料，由於在網路上發送更少的資料而進一步增強性能。

線上壓縮也完美地和糾刪碼配合使用。

對於離線壓縮，所有新的寫 I/O 按照正常 DSF 的 I/O 通道沒有壓縮地寫入。當壓縮延遲（可配置）到達後，資料才有資格被壓縮。壓縮可以發生於 Extent Store 的任何地方。離線壓縮使用 Curator 的 MapReduce 框架，所有節點將參與壓縮任務。壓縮任務將被 Chronos 控制。

下圖展示了一個離線壓縮和 DSF 寫 I/O 通道如何交互的例子：

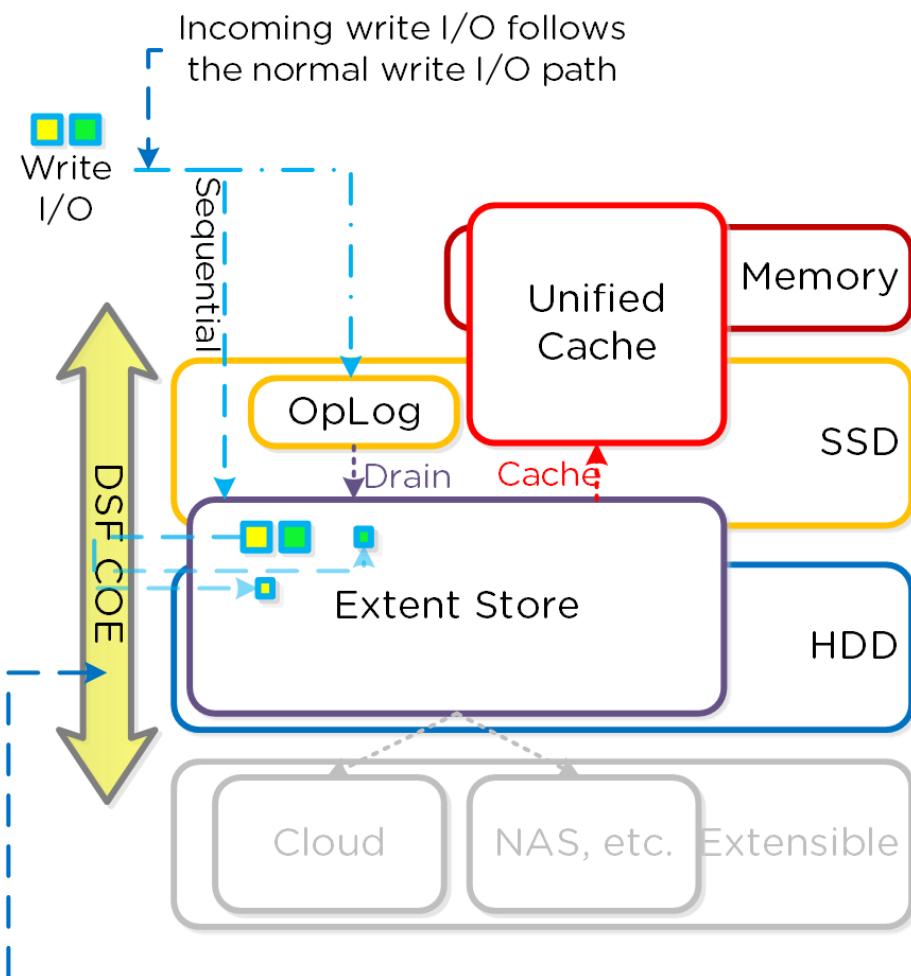


圖. 離線壓縮 I/O 路徑

對於讀 I/O, 資料首先在記憶體解壓縮之後 I/O 被讀取。

你可以從 Prism 上，在存儲>儀錶盤頁面上看到當前壓縮率。

4.3.7.3 彈性消重引擎

你可以通過觀看下面視頻來說明理解：<https://youtu.be/C-rp13cDpNw>

彈性消重引擎是 DSF 中一個基於軟體的特性，它允許在容量層面（HDD）和性能層面（SSD/記憶體）對資料進行消重操作。順序的資料流程在寫入（ingest）時，按照16KB 細微性進行 SHA-1 的散列（hash）運算標記指紋。指紋標記操作僅在資料塊被寫入（ingest）的時候進行，並作為該資料塊的中繼資料資訊的一部分被持久保存。注意：最初採用 4KB 細微性進行指紋標記，然而經過測試， 16KB 細微性的指紋標記操作能在消重能力和減少中繼資料(metadata)開銷之間取得最好的平衡。當已消重資料進入unified cache 之後，將以 4KB 細微性進行消重。

傳統消重操作需要在後臺進行資料塊掃描，資料被重新讀取並開銷大量系統資源， Nutanix 在寫入時執行線上的指紋標記操作。消重時，外部存儲上重複資料塊可以被直接刪除，而無需重新讀取和計算指紋。

為了使中繼資料更加高效率，指紋標記參考計數被監控來跟蹤可消重能力。參考計數低的指紋計數將被丟棄以減少中繼資料的超載。為最小化碎片，完全 extents 是用於在容量層面的消重首選。

下圖顯示了彈性消重引擎如何擴展和處理本地虛擬機器 I/O 請求的：

專家提示：

對基本影像（Base images）使用性能層面的消重會利用內容緩存的優勢。
（你可以手工對它們進行指紋標記，使用 vdisk_manipulator）。

當使用 Hyper-V 時（由於 ODX 實行完全資料拷貝），或者進行跨容器的克隆時（通常不建議，因為最好是一個容器），使用容量層面對 P2V/V2V 消重。

在大部分其它案例，壓縮是節省容量最大的，應該建議使用。

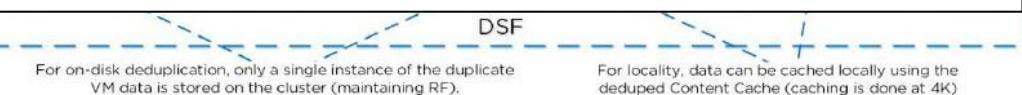


圖.彈性消重引擎 - 擴展性

指紋標記操作是在資料寫入大小為 64KB 或更大時發生，Intel 的加速器能使得 SHA-1 的計算增加的額外 CPU 開銷最小。如果資料不是持續寫入（eg. 小 IO 操作），將不進行指紋標記操作，此時指紋標記操作將在後臺發生。彈性消重操作不僅發生在磁片層面（HDD），並且也發生在高性能的存儲層面（SSD/Memory）。當確定了哪些重複資料後，即基於相同指紋的多份資料，一個後臺進程會使用 DSF 的 MapReduce 框架（curator）來移除重複資料。

由於標記指紋的資料已經被讀取，會被保存在 DSF 的 Unified Cache 中（這是 multi-tier/pool cache）。任何後續對於同樣指紋的資料請求將直接從 Cache 層面回應。要瞭解統一緩存和池架構，請參考 I/O 路徑概覽章節相關內容。

專家提示：

指紋標識 vDisk Offsets

在 4.5 之前，只有 vDisk 的前 12GB 可以被指紋標識，這是用來維護一個較小的中繼資料指紋標識並且由於 OS 通常是最普通的資料。4.5 之後，增加到 24GB，因為中繼資料更加高效。

下圖顯示了彈性消重操作如何優化 I/O 操作：

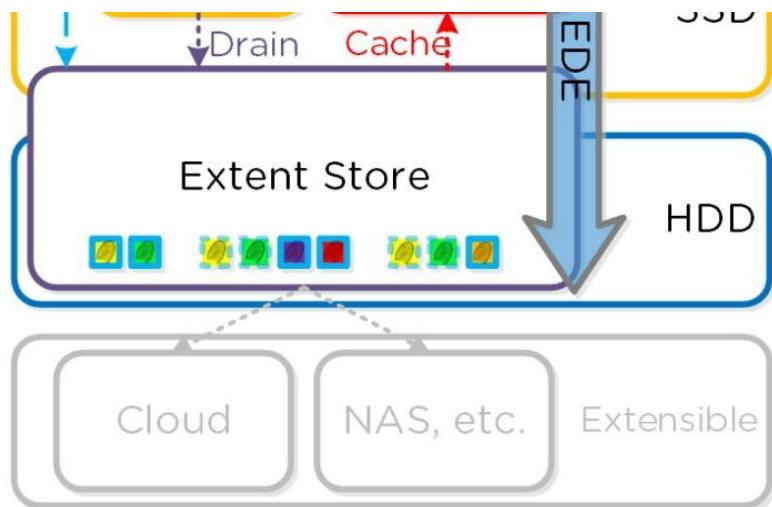


圖 . 彈性消重引擎 I/O 路徑

你可以從 Prism 上的存儲>面板頁查看當前的消重率。

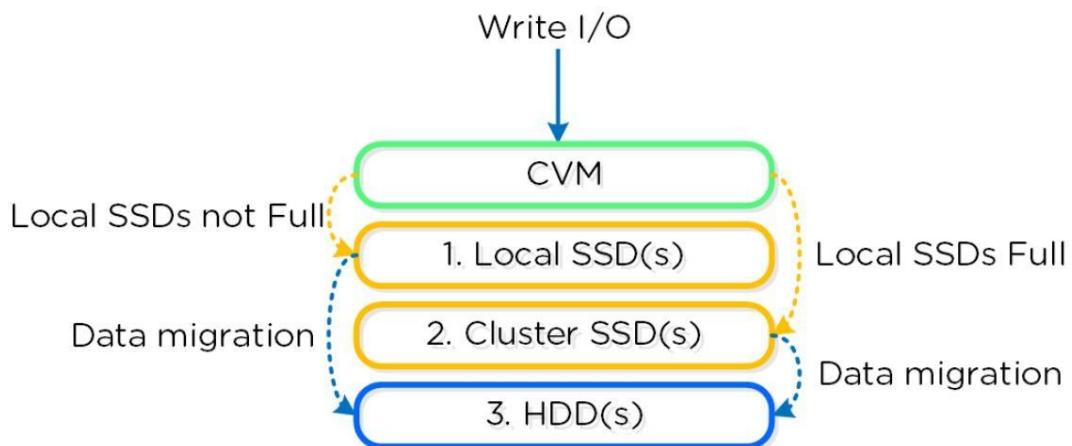
專家提示：消重+壓縮

在 4.5 版本，消重和壓縮都可以在一個容器裡啟用，但是，除非資料是可以被消重的（條件在之前的章節裡解析過），堅持使用壓縮。

4.3.8 存儲分層和優先順序

在磁片平衡章節我們談到了如何將集群內所有節點的磁片容量作為存儲池統一管理，並且通過 ILM 實現熱資料當地語系化。簡單而言，磁片的分層是實現在集群內所有節點的 SSD 和 HDD 上，並且由 ILM 負責觸發資料移轉。本地節點的 SSD 層總是最優先 級的，負責所有本地虛擬機器 I/O 的讀寫操作。並且還可以使用集群內所有其他節點的 SSD，因為 SSD 層總是能提供最好的讀寫性能，並且在混合陣列中尤為重要。

下圖顯示了熱分層優先順序順序：



*NOTE: Sequential IO can be configured to bypass SSD and be directly written to the HDD tier.

圖. DSF 層級優先順序

特定類型的存儲資源 (eg. SSD, HDD, 等) 在集群範圍內被池化統一管理，形成集群範圍的存儲分層。這意味著集群內任何節點都可以使用到整個存儲分層，無論這一存儲資源在本地還是在其他節點上。

下圖顯示的是池化後的集群分層示意圖：

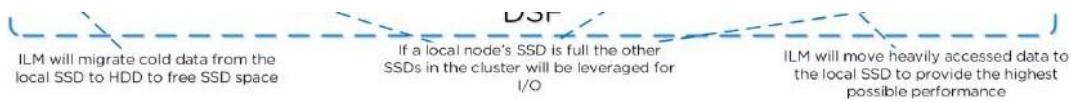


圖. DSF 集群範圍內的分層

一個經常被問到的問題就是：如果節點本地的 **SSD** 被用滿之後，性能是否會下降？之前我們提到使用磁片平衡功能會保證集群範圍內所有磁片的利用率基本一致。那麼當本地 **SSD** 利用率非常高時，磁片平衡功能會自動遷移 **SSD** 中最冷的資料到集群內另一個節點的 **SSD** 上。這將釋放本地 **SSD** 的空間，資料會繼續寫到本地 **SSD** 上，而不會因為通過網路寫到另一個節點的 **SSD** 上而導致性能下降。一個關鍵點需要提及的是所有 **CVM** 和 **SSD** 都參與這種遠端 **IO** 操作，可以消除潛在的瓶頸，並且通過執行遠端 **IO** 操作可以自愈一些命中率的問題。



圖. DSF 集群內的分層均衡

另一種情況是當存儲利用率到達一定的閾值

(**curator_tier_usage_ilm_threshold_percent**, 默認是 75)，DSF ILM 將介入並作為 **Curator** 框架的一個工作，並將資料從 **SSD** 上遷到 **HDD** 上。這會清空超過上述閾值的空間，或者通過參數 (**curator_tier_free_up_percent_by_ilm**, 默認是 15) 清空資料空間，這兩個閾值取較大的那個。

遷移資料是使用最後存取時間來判斷。例如當 **SSD** 利用率到 95%，那麼將有 20% 的資料會被移動到 **HDD** (95% -> 75%)，如果 **SSD** 利用率時 80%，則只有 15% 的資料會被移動到 **HDD**。

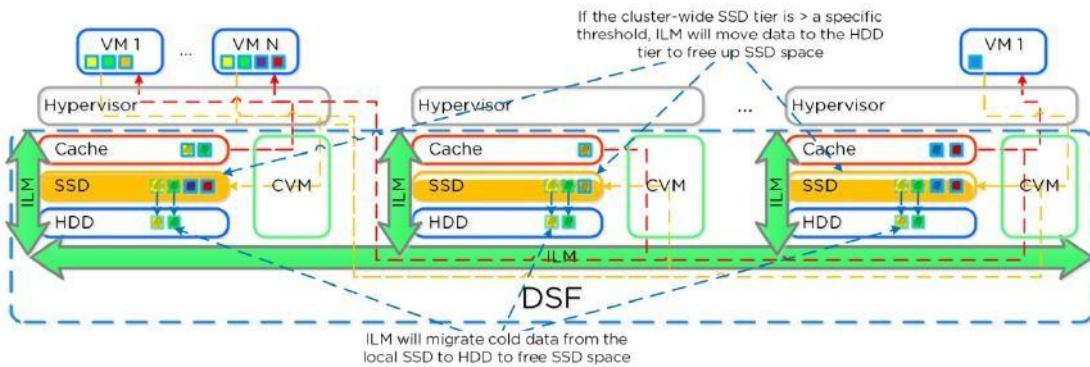


圖. DSF 層級 ILM

DSF 的 ILM 也會持續監控 I/O 類型資料分佈和必要的資料移轉（向下/向上），使最熱的數據當地語系化。

4.3.9 磁片平衡

你可以通過觀看下面視頻來說明理解：<https://youtu.be/atbkgrmpVNo>

DSF 被設計成為非常動態的平臺，可以適用於不同工作負載的應用，並且允許混合節點類型：例如將計算密集型（3050 系列）和存儲密集型（60X0 系列）混合在一個集群中。對於集群內部磁片容量大小不同的，確保資料一致的分佈非常重要。

DSF 有自帶的稱為磁片平衡的技術，用來確保資料一致的分佈在集群內部各節點上。磁片平衡功能與各節點的本地磁片利用率和內置的 **DSF ILM**（資料生命週期管理）一同工作。它的目標是一旦利用率超出設定閾值時，使得所有節點的磁片利用率大致相等。

注意：磁片平衡作業由 **Curator** 處理，後者對 **Primary I / O** (UVM I / O) 和 **background I / O** 具有不同的優先順序佇列（例如，磁片平衡）。這樣做是為了確保磁片平衡或任何其他後臺活動不會影響前端延遲/性能。在這種情況下，工作的任務將交給 **Chronos**，後者將限制/控制任務的執行。此外，移動只會在同一層內完成以實現磁片平衡。例如，如果我的資料在 **HDD** 層中出現不平衡的情況，則我將在同一層 (**HDD**) 的節點之間移動。

下圖顯示了一個混合集群（3050 系列+6000 系列）的不平衡狀態：



Disk balancing will move the coldest data to other nodes in the cluster to ensure uniform distribution

圖. 磁片均衡 - 非平衡狀態

磁片平衡功能使用 DSF 的 Curator 框架作為調度進程，當磁片閾值被觸發時，在後臺自動運行（例如，當本地節點利用率超過 n% 後自動運行）。當資料不平衡時，Curator 會決定哪些資料移動到哪裡，並自動生成 MapReduce 任務分發到各節點執行。例如，集群各節點同樣都是相同型號（例如，3050 系列），每個節點的利用率應該基本一致。

然而，當一些 VM 寫了過多的資料，導致該 VM 資料快速增長，使得該節點磁片利用率高於其他節點。此時磁片平衡功能可以將該節點上最冷的資料移動到其他節點上，保證所有節點磁片利用率達到基本平衡的狀態。例如，集群各節點時不同型號（例如，3050+6020/50/70 系列），或者一些節點僅作為存儲節點使用（不運行 VM）時，可能會觸發磁片平衡功能。

下圖顯示了一個混合集群在磁片平衡後的“平衡”狀態：

This process is done both during runtime with node/disk placement as well as a background Curator process

圖. 磁片均衡 - 平衡狀態



在一些情形下，客戶可能希望將一個節點只作為存儲使用，則這個節點上將沒有虛擬機器運行，它的主要功能作為大型存放區設備使用。在這種情況下，節點上所有記憶體都可以給到 CVM 使用，來提供更大的讀緩存。

下圖顯示了混合集群中包含僅存儲節點，並使用“磁片平衡”功能後資料從活動虛擬機器節點移動到它上的情況：

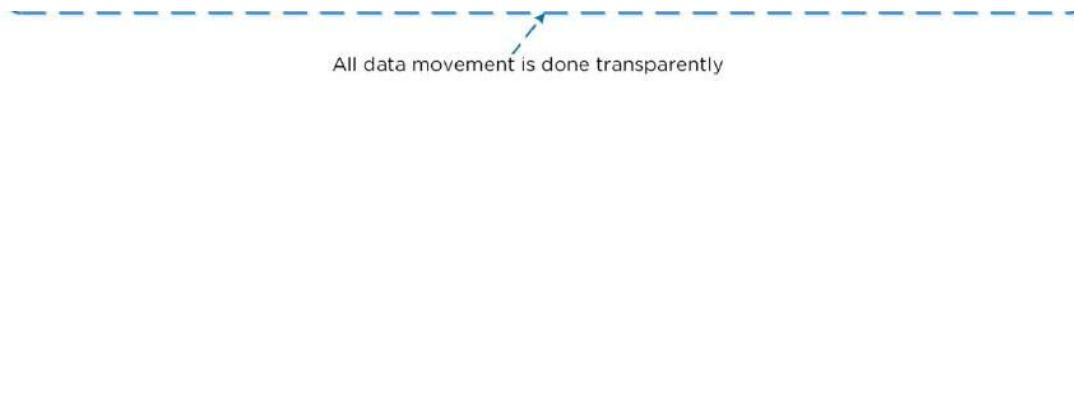


圖. 磁片均衡 - 單純的存儲節點

4.3.10 快照和克隆

你可以通過觀看下面視頻來說明理解：<https://youtu.be/uK5wWR44UYE>

DSF 內置支持快照和克隆的卸載(offload)，能夠利用 VAAI, ODX, ncli, REST, Prism 等進行快照和克隆。快照和克隆均利用最有效和高效的寫時重定向（redirect-on-write）演算法。正如在上述資料結構章節解析的，一個虛擬機構成的檔（vmdk/vhdx）在 Nutanix 平臺是 vDisks。

vDisk 是由 extend（邏輯上連續的資料塊）構成，extend 存放在 extend 組裡，extend 組是物理上連續的資料，在存放裝置裡以檔形式存放。當快照或者克隆發生時，基礎 vDisk 被標識為不可改變的，創建另外一個 vDisk 為可讀寫。這時候，兩個 vDisk 都有相同的資料塊映射，它是 vDisk 到相應 extend 的中繼資料映射。傳統的方法需要遍歷快照鏈(增加讀延遲)，而現在每個 vDisk 有它自己的資料塊映射。這樣就消除了常見的很深的快照鏈的超載，允許你持續地進行快照而不影響性能。

下圖顯示了快照是如何進行工作的例子：



圖. 快照塊映射實例

當一個 vDisk 的快照和克隆繼續進行快照和克隆時，採用相同的方法：

Extents Extents

圖.多重快照塊映射和新的寫入

採用相同的方法進行虛擬機器和 vDisk 的快照和克隆。當一個虛擬機器或 vDisk 進行克隆時，當前的塊映射被鎖定，克隆被創建出來。這些更新僅僅是中繼資料，沒有 I/O 實際發生。相同的方法應用到克隆的克隆；之前克隆的虛擬機器充當被克隆的“基礎vDisk”，資料塊映射被鎖定，2 個克隆被創建：一個給正在被克隆的虛擬機器，另外一個給新的克隆。

它們均繼承了之前的資料塊映射，任何新的寫入和更新將會發生在它們單獨的資料塊映射裡。

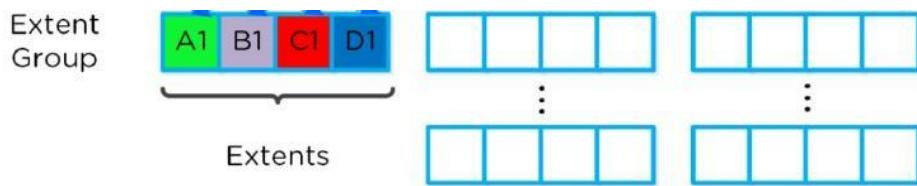


圖.多重克隆的塊映射

如之前說過，每個 **VM/vDisk** 有它自己的單獨的資料塊映射。所以在上述例子裡，所有從基礎虛擬機器來的克隆將會有它們自己的資料塊映射，所有寫和更新將會發生在那裡。

如果發生覆寫，資料將進入新的 **extent / extent group**。例如，假如在 **extent e1** 中偏移量 **o1** 處的現有資料被覆寫，**Stargate** 將創建一個新的 **extent e2** 並跟蹤新資料在 **extent e2** 的偏移量 **o2** 寫入。**Vblock** 映射將跟蹤到位元組級別。

下圖顯示了這些看起來是什麼樣的例子：



Extents

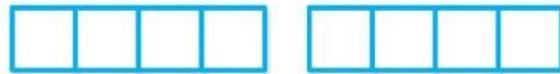


圖 . 克隆塊映射 - 新寫入

任何後續 VM/vDisk 克隆或快照將會引起原始資料塊映射被鎖定，並將產生一個新的讀寫訪問。

4.3.11 網路及 I/O

關於視頻解說，你可以查看以下連結：(https://youtu.be/Bz37Eu_TgxY)

Nutanix 平臺的內部節點間無背板連結，而是依賴於標準的 10GbE 網路進行通訊。運行於 Nutanix 平臺上的虛擬機器的所有存儲 I/O 都在專用私有網路裡由虛擬化監控程序 (HyperVisor) 處理。虛擬化層接收 I/O 請求，隨後將其轉發給本地 CVM 的私網 IP。此 CVM 隨後將通過外部 IP 及外部的 10GbE 網路把資料遠端複製至其他CVM 中。對於讀請求，絕大部分都將由本地 CVM 提供服務，而無需通過外部的 10GbE 網路。這就意味著，只有 DFS 的遠端複製和 VM 的網路流量需要用到 10GbE 網路，除非本地 CVM 容機或跨節點訪問遠端資料（如 vMotion 後）。當然，集群的其他一些情況也會臨時使用到 10GbE 網路，例如：磁片的負載均衡等。

以下是 VM 使用到的 I/O 路徑的示意圖，包含內部、外部 10GbE 網路：

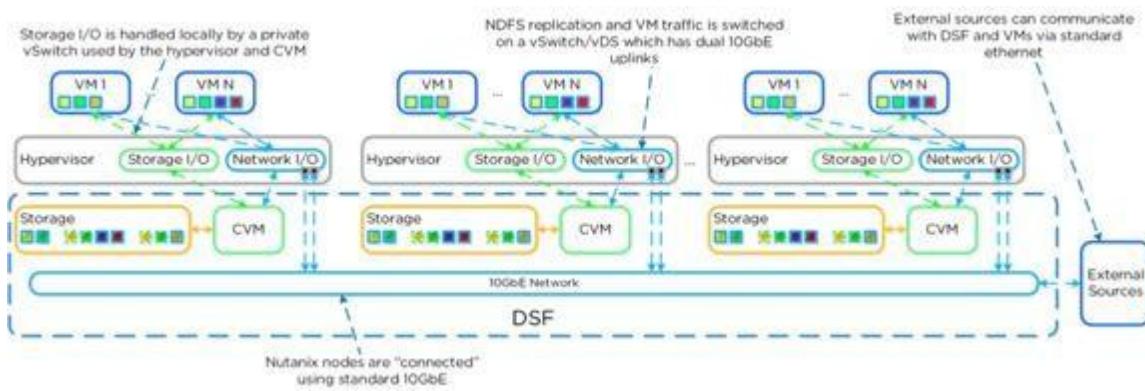


圖. DSF 網路

4.3.12 數據當地語系化 Data Locality

關於視頻解說，你可以查看以下連結：<https://youtu.be/ocLD5nBbUTU>

I/O 和資料的當地語系化（**data locality**），是 Nutanix 超融合平臺強勁性能的關鍵所在。正如之前 I/O 路徑（I/O Path）章節所述，所有的讀、寫 I/O 請求都藉由 VM 的所在節點的本地 CVM 所回應處理。VM 的資料都將由本地的 CVM 及其所管理的本地磁片提供服務。當 VM 由一個節點遷移至另一個節點時（或者發生 HA 切換），此 VM 的資料又將由現在所在節點中的本地 CVM 提供服務。當讀取舊的資料（存儲在之前節點的 CVM 中）時，I/O 請求將通過本地 CVM 轉發至遠端 CVM。所有的寫 I/O 都將在本地 CVM 中完成。DFS 檢測到 I/O 請求落在其他節點時，將在後臺自動將資料移動到本地節點中，從而讓所有的讀 I/O 由本地提供服務。資料僅在被讀取到才進行搬遷，進而避免過大的網路壓力。

資料當地語系化有兩種主要形式：

- 緩存當地語系化
 - 將遠端資料拉入本地 Stargate 的統一緩存。這是在 4K 細微性下完成的。
 - 對於沒有本機複本的實例，請求將轉發到包含副本的 Stargate，後者將返回資料，本地 Stargate 將此資料存儲在本地，然後返回 I/O。對該資料的所有後續請求將從緩存中返回。
- Extent Group (egroup) 當地語系化
 - 遷移要存儲在本地 Stargate 的擴展存儲區中的 vDisk 擴展區組 (egroup)。
 - 如果副本 egroup 已經在本地，則無需移動。
 - 在這種情況下，在滿足某些 I/O 閾值後，實際副本 egroup 將重新定位。我們不會自動重新當地語系化/遷移 egroup，以確保我們有效利用網路。
 - 對於啟用 AES 的 egroup，在副本不在本地且滿足模式的情況下，也會發生相同的水準遷移。

以下展示了資料是如何隨著 VM 而遷移的：

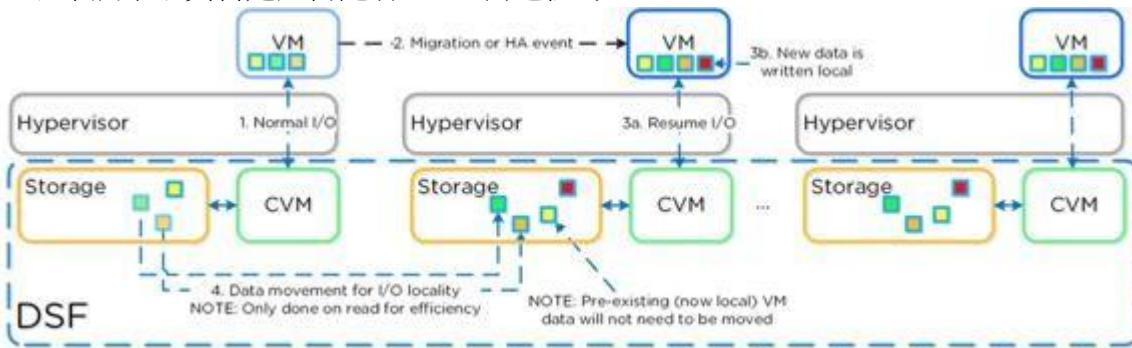


圖. 數據當地語系化

資料移轉條件：

Cache 當地語系化基於 vDisk 歸屬即時發生，當一個 vDisk/VM 從一個節點移動到另一個，vDisk 的歸屬被移交到本地 CVM，之後資料在本地 Unified Cache 裡存儲。中間過程 Cache 將被歸屬主機暫停（現在是遠端主機）。當它看見遠端I/O 超過 300 秒，之前主機的 Stargate 將放棄 vDisk 標記，由本地 Stargate 接管。緩存 vDisk 的資料需要得到所屬權利，由此 Cache 一致性得到增強，

Extend 當地語系化是一個取樣操作，當滿足以下條件時將搬遷 Extent Group：10分鐘內被觸發 3 次隨機讀或者 10 次順序讀，其中 10 秒內的多次讀取被視為 1 次觸發。

4.3.13 影子克隆 Shadow Clones

關於視頻解說，你可以查看以下連結：<https://youtu.be/oqfFDMyQFJg>

Acropolis DSF 有一項功能稱之為“Shadow Clones”，允許在大量併發讀取時分散式的緩存特定的 vDisk 或 VM 資料。最好的例子莫過於在 VDI 部署中，大量的連結克隆（Linked Clones）需要讀取模版虛擬機器（A central master or ‘Base VM’）。在Vmware View 中，這被稱之為磁片副本（replica disk），並被所有連結克隆所讀

取。在 XenDesktop 中，這被稱之為 MCS 主虛擬機器（MCS Master VM）。這也可以被用於其他的大量併發讀取的環境中（例如：部署虛擬伺服器等）。資料和 I/O 當地語系化是將 VM 性能最大化，以及 DSF 的架構特性的關鍵之所在。

關於 Shadow Clones，DSF 將監控 vDisk 的“當地語系化”訪問趨勢。如果有超過 2 個以上 CVM（包括本地的 CVM）同時訪問，且全部都是讀請求，該 vDisk 將被標記為只 讀。一旦 vDisk 被標記為唯讀，則該 vDisk 可被緩存至所有 CVM 的本地，實現本地的資料讀取（或可稱之為基於 vDisk 的 Shadow Clones）。這樣就可以讓所有的 VM

在本地就可以讀取模版虛擬的 vDisk 了。在 VDI 中，這就意味著所有的磁片副本可以被緩存到所有節點的本地，所有的讀請求將由本地節點給予回應。注意：資料將在讀取後執行搬遷，因此不會大量增加網路負載，並且可以大幅提升緩存的利用率。如果模版虛擬機器被更改，則 Shadow Clones 將停止，並丟棄。Shadow Clones 默認被開啟（如 NOS4.0.2），你可以通過以下 NCLI 命令列開啟或關閉此功能：`ncli cluster edit-params enable-shadow-clones=<true/false>`

下圖展示的 Shadow Clones 是如何工作及允許分散式緩存的：

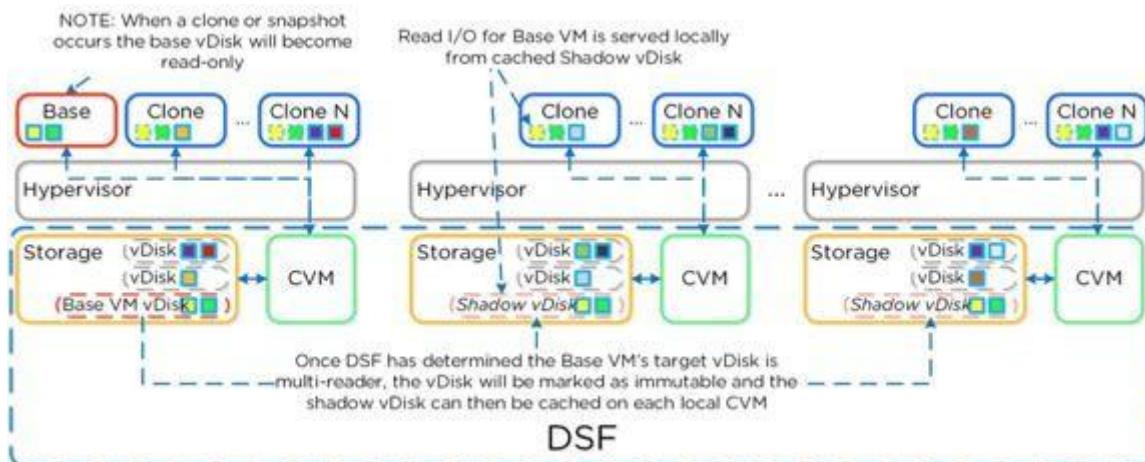


圖.影子克隆

4.3.14 存儲層和監控

Nutanix 平臺監控著存儲中的多個層面，從 VM 或 Guest OS 到物理磁片。瞭解各個層面以及它們之間的關係對於後續的監控、運維、操作等是很重要的。

下圖顯示各個層面相關的監控事項及監控的顆粒度等：



圖. 存儲層級



虛擬機器層

- 主要角色：通過虛擬化層獲取 VM 的狀態參數
- 描述：通過虛擬化層直接獲取 VM 或 Guest 的狀態參數，標示出 VM 當前的性能，以及應用系統當前的 I/O 性能
- 使用場景：故障定位或 VM 性能分析

虛擬管理器層

- 主要角色：虛擬化層的狀態參數
- 描述：通過虛擬化層獲取當前的最為精准的狀態參數。這些資料包含整個群集或其中任一節點的狀態。這一層面上可以提供各個平臺中的性能參數，可被用於更多的分析場景。當然，你也可以通過監控不同的顆粒度，獲取更加精准的指標。這其中就包含 Nutanix CVM 的緩存命中率
- 使用場景：提供更有價值的詳細的性能指標

控制器層

- 主要角色：Nutanix 控制器的狀態參數
- 描述：此狀態參數由 Nutanix 控制器虛擬機器（例如：通過 Stargate 2009 埠）直接提供，它將標示出前端採用了 NFS/SMB/iSCSI 中的哪種連結的方式，以及後端發生了哪些操作（例如：ILM、磁片負載均衡等）。你可以查看單個控制器虛擬機器或整個控制器集群的相關參數。這些狀態參數中部分資料將與虛擬化層的提供的資料相吻合，另外，資料中還將包括後端的操作（例如：ILM、磁片負載均衡等），當然這其中就包含 Nutanix CVM 的緩存命中率。在某些情況下，IOPS 的指標可能會有差異，因為 NFS/SMB/iSCSI 用戶端會將大的 IO 進行拆分。但它們表現出來的頻寬應該是一致的
- 使用場景：類似於 HyperVisor，卻能夠更好監控後端操作

磁片層

- 主要角色：磁片設備的狀態參數
- 描述：此狀態參數由物理磁設備直接提供（通過 CVM），標示出了後端的操作。其中包括 I/O 在磁片操作時命中 Oplog 或者 Extend Store。可以監控到單塊磁片、某個節點的磁片，或集群中的全部磁片。一般情況下，對磁片的操作包括 IO 的寫入及未在緩存中命中的讀操作。任何在緩存中命中的讀操作，都將不計入磁片的監控資料中
- 使用場景：可以查看有多少 IO 操作落到了緩存或磁片中

保存期限

以上狀態資訊，在本地的 Prism 中僅保存 90 天。如使用 Prism Central 和 Insights，則可以長期保存（如果容量充足的話）

4.4 服務

4.4.1 Nutanix Guest Tools (NGT)

Nutanix Guest Tools(NGT) 是一個安裝在客戶機作業系統上的軟體代理（類似於 vmtools），它使得虛擬機器能夠使用 Nutanix 平臺的一些高級功能。

Nutanix Guest Tools 包括安裝在虛擬機器中的 NGT installer 和 Guest Tools 程式框架。通過 Guest Tools 程式框架在代理程式和 Nutanix 平臺之間進行協調。

Nutanix Guest Tools(NGT)通過 NGT Installer 安裝程式進行安裝配置，NGT Installer 安裝程式包括如下元件：

- 客戶機代理服務；
- 自助服務恢復 Self-service Restore(SSR)，也稱為基於命令列的檔級自服務恢復 File-level Restore(FLR)
- 虛擬機器移動性驅動（AHV 的 VirtIO 驅動）；
- 針對 Windows 虛擬機器的 VSS(volume shadow service 卷影服務)代理和硬體介面服務（Hardware Provider）；
- 針對 Linux 虛擬機器的應用一致性快照支持（通過腳本實現靜默）；這個程式框架還包括一些高級元件：
 - 客戶機工具元件服務（Guest Tools Service）
 - 作為在 Acropolis、Nutanix 服務及客戶機代理之間的閘道。分佈在集群內的各個 CVM 上，該集群運行了在當前 Prism leader（託管主機集群 vIP-虛擬 IP）上通過選舉出來的 NGT Master。
 - 客戶機代理（Guest Agent）
 - 代理程式與相關服務作為 NGT 安裝進程的一部分，部署在客戶機作業系統中。處理所有本機服務（例如 VSS、SSR 等）與前述客戶機工具元件服務的交互。

下圖顯示了元件的高級映射：

The Guest Tools Service will listen on port 2074 on the cluster VIP which will be hosted by a CVM

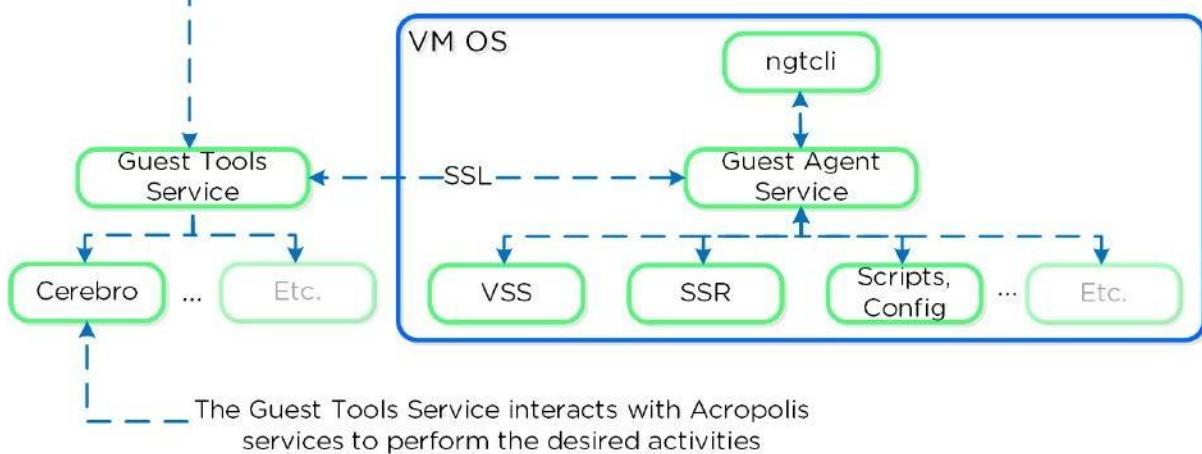


圖.Guest Tools 映射

客戶機工具元件服務 (Guest Tools Service)

客戶機工具元件服務由兩個主要角色組成：

- NGT Master

處理由 NGT Proxy 發來的請求並與 Acropolis 組件交互。每個集群動態選舉得出一個獨立的 NGT Master，如當前 Master 失效，則將再選舉出一個新的。該服務的內部監聽埠為 2073。

- NGT Proxy

運行在每個 CVM 上，並轉發請求給 NGT Master 以執行需要的活動。由當前作為 Prism Leader (託管集群虛擬 IP) 的 CVM 虛擬機器與客戶機代理程式進行通信。監聽的外部埠為 2074。

當前 NGT Master

您可以用下面命令列找到 NGT Master 角色所在的 CVM (在 CVM 運行)

```
nutanix_guest_tools_cli get_master_location
```

下圖顯示了高級別的角色映射：

Guest Tools Service

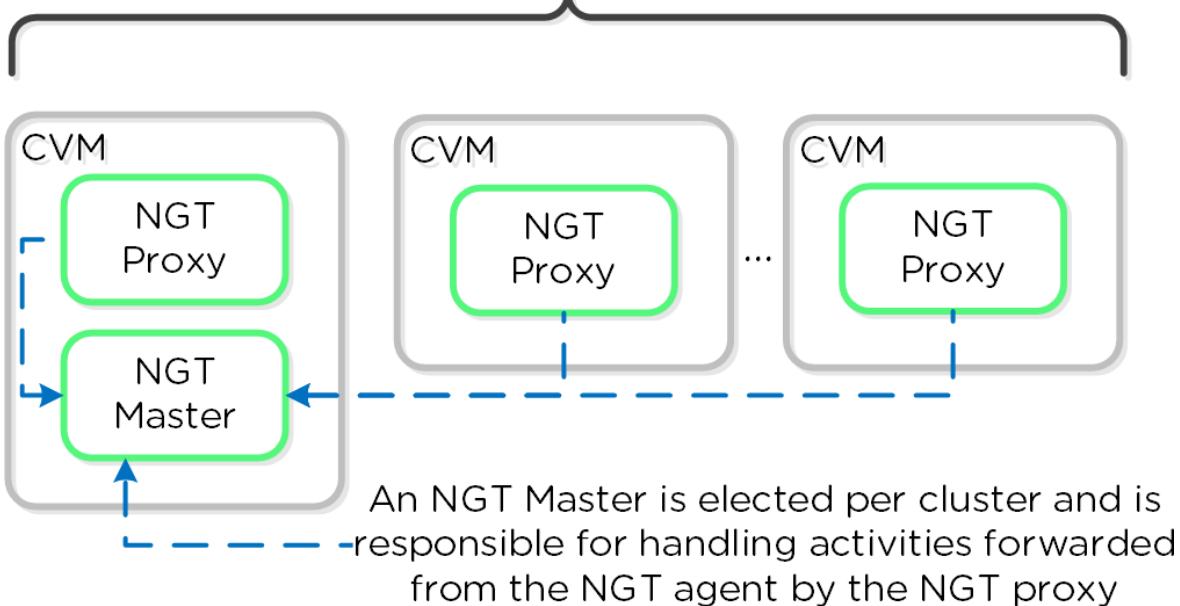


圖. 客戶機工具服務

客戶機代理程式 (Guest Agent)

客戶機代理程式由以下高級別元件按之前已提到過的優先順序組成：

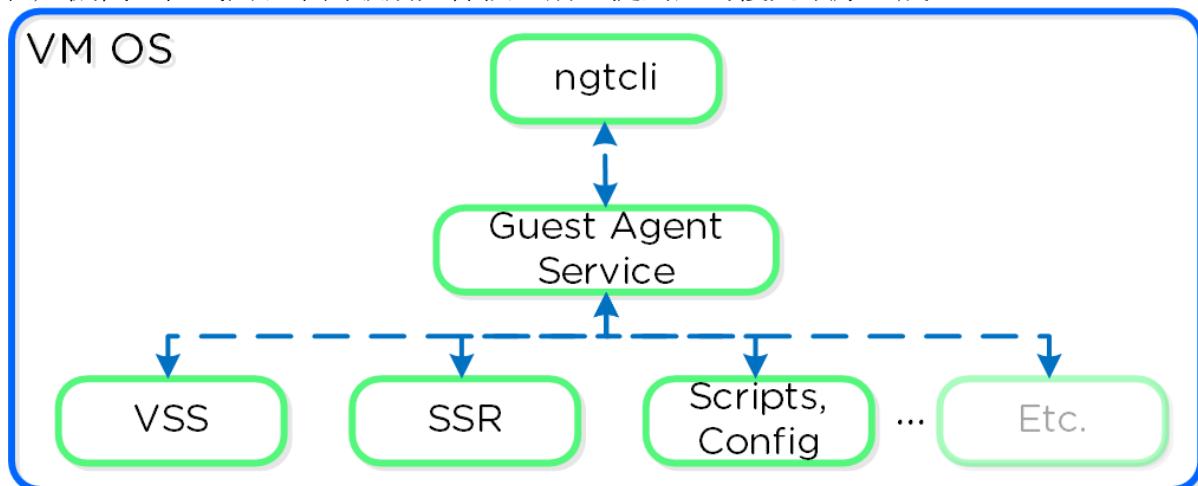


圖. 客戶機代理程式

通信與安全



客戶機代理服務與客戶機工具元件服務之間的通信通過 Nutanix 集群 IP 的 SSL 連結進行。當 Nutanix 集群元件和 UVM 的部署不在同一網路時，需要確認以下配置是可行的：

- 確認 UVM 網路到集群 IP 的路由可達
或者
- 在 UVM 網路中創建一個防火牆規則（以及相關的 NAT 規則），允許與集群 IP 在 2074 埠上進行通信。（優選方案）

客戶機工具服務還要充當 CA 認證，負責在每一個已運行 Nutanix Guest Tool 的 UVM 上生成認證鑰匙對。證書被嵌入 Nutanix Guest Tool 的 ISO 安裝檔並作為安裝進程的一部分被安裝在 UVM 上。

NGT 代理程式的安裝

NGT 代理程式的安裝可通過 Prism 或通過命令列/腳本（ ncli/REST/PowerShell）。通過 Prism 安裝 NGT，在 Prism 介面導航到“VM”頁面，選擇一個虛擬機器安裝 NGT 代理程式，然後點擊“Enable NGT”

VM NAME	HOST	IP ADDRESSES
WIN2K12BASE	NTNX-BEAST-8	10.3.145.222

Summary > WIN2K12BASE → Enable NGT

圖. 為虛擬機器啟動 NGT

在彈出的提示欄上點擊“Yes”以繼續 NGT 的安裝；



NGT feature will be enabled for this VM and the
NGT CD-ROM image will be mounted. Do you
want to continue?



圖. 啟動 NGT 安裝的提示欄

安裝 NGT 的虛擬機器必須配置一個 CD-ROM 用以掛載 NGT 安裝套裝軟體，如下圖所示：

Disks

+ Add new disk

TYPE	ADDRESS	PARAMETERS	Actions
DISK	scsi.0	SIZE=74.51GiB; CONTAINER=KVM...	edit · X
CDROM	ide.0	SIZE=0.08GiB; CONTAINER=KVM...	▲ · edit · X

圖. 啟動 NGT 安裝 – 光碟機設備

在虛擬機器作業系統中將能看到 包含 NGT 安裝套裝軟體的光碟機設備：



圖. 啟動 NGT 安裝 – 虛擬機器作業系統中的光碟機設備

滑鼠按兩下上圖光碟機設備圖示以開始安裝進程。



靜態安裝

您可以通過運行下列命令（從 CD-ROM）執行一個靜態安裝：



在出現的提示欄中點擊接受軟體授權合約以完成 NGT 套裝軟體的安裝：

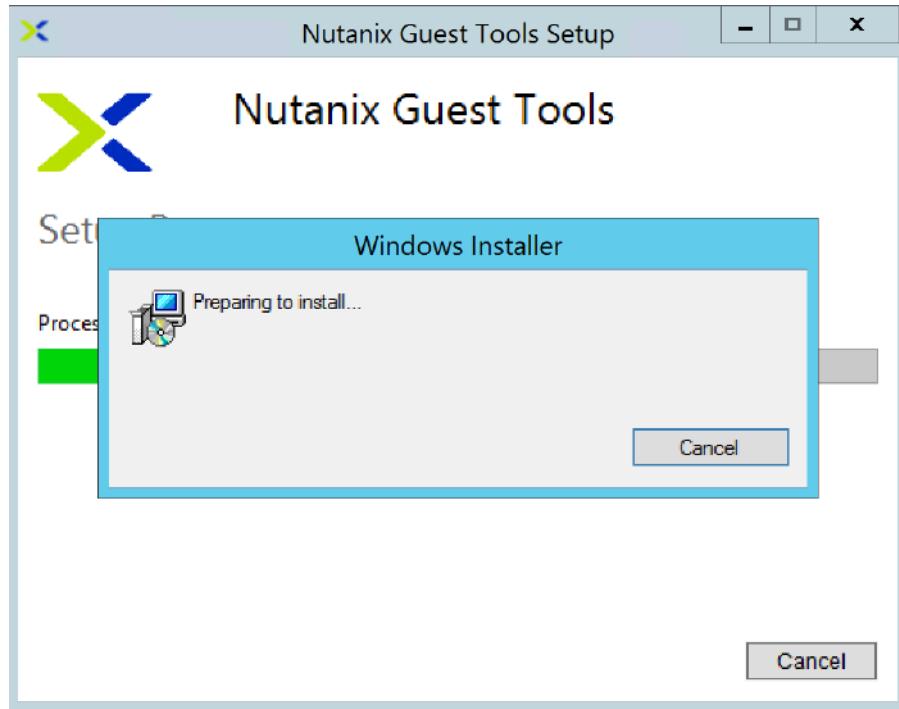


圖. 啟動 NGT 安裝 – 開始安裝

作為安裝進程的一部分，Python、PyWin 和 Nutanix Mobility（提供跨虛擬化平臺的互換能力）驅動將同時被安裝。

安裝結束後，需要重啟虛擬機器作業系統。

成功完成安裝和重啟任務後，您將能看到以下條目出現在控制台的“程式與功能”專案中。

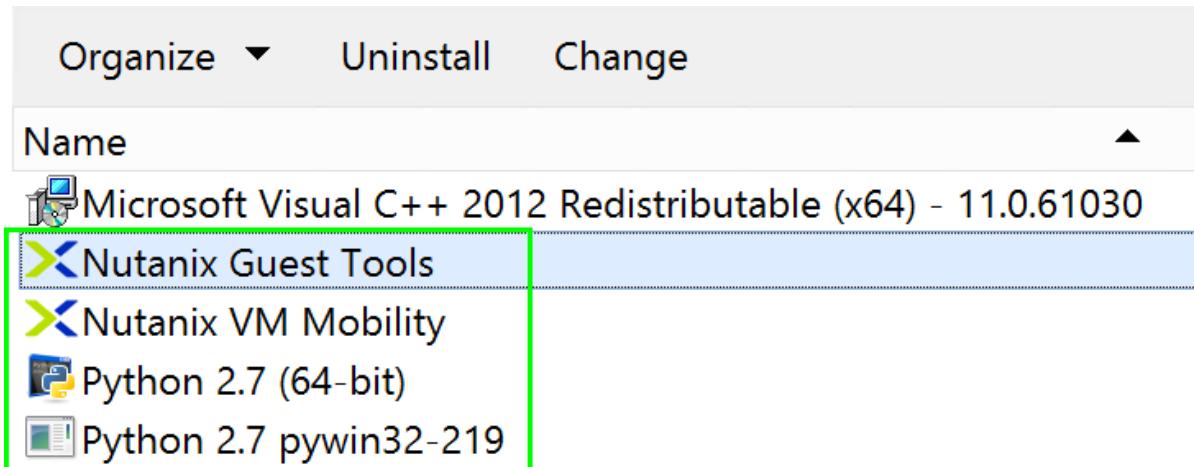


圖. 啟動 NGT 安裝 – 已安裝的程式

在服務專案中可以看到 NGT Agent 服務和 VSS Hardware Provider 服務已經運行：

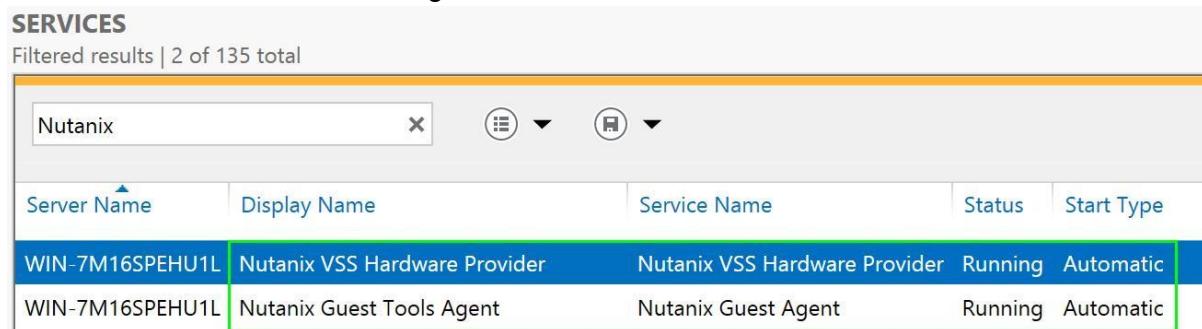


圖. 啟動 NGT 安裝 – 安裝後啟動的服務

此時，NGT 已經完成部署並能夠被使用了。

備註：

批量 NGT 部署

與其在每一個虛擬機器上分別安裝 NGT，更好的方式是在您的基礎作業系統鏡像中嵌入並部署 NGT。

可通過以下步驟在作業系統基礎鏡像中嵌入並部署 NGT：

1. 在原版虛擬機器上安裝 NGT 並確認能夠正常通信；
2. 基於該母版虛擬機器克隆新的虛擬機器；
3. 在每個克隆虛擬機器上掛載 NGT 的 ISO 檔 (因需要得到新的認證鑰匙對)
 - 命令列示例：ncli ngt mount vm-id=<CLONE_ID> OR via Prism



- 後續提供自動化的方式；
- 4. 啟動克隆虛擬機器

當克隆生成的虛擬機器啟動時，它將檢測到新的 NGT ISO 安裝鏡像並拷貝相關配置

4.4.2 OS 定制化

Nutanix 借助於 CloudInit 和 Sysprep 提供原生的 OS 定制化能力。CloudInit 是一個控制 Linux Cloud Server 引導的套裝軟體，它可以提前初始化和定制化 Linux Instance。Sysprep 是 Windows 作業系統的 OS 定制化工具。

常見的使用如下：

- 設置機器名
- 安裝套裝軟體
- 添加用戶／金鑰管理
- 定制化腳本

支援的配置

解決方案適用於運行在AHV 上的Linux 客戶機，支援以下版本（部分清單，完整的支援清單詳見相關文檔）：

- Hypervisors：
 - AHV
- 作業系統：
 - Linux - 大部分當前常見版本
 - Windows - 大部分當前常見版本

前提條件



使用 CloudInit 需要滿足以下條件：

- CloudInit 已經在 Linux 系統中安裝完成。

Windows 默認已經安裝了 Sysprep

套裝軟體安裝

CloudInit 如果不可用，可以通過以下命令進行安裝：

- 基於 RedHat 的 CentOS 和 RHEL：

```
yum -y install CloudInit
```

- 基於 Debian 的 Ubuntu：

```
apt-get -y update; apt-get -y install CloudInit
```

Windows 基礎安裝已經包含了 Sysprep

鏡像定制化

為了進行 OS 定制化，在創建或克隆虛擬機器時，在 Prism 或 REST API 中要選擇Custom Script 核取方塊並在 Script 輸入框中輸入定制化的腳本。



Custom Script
Provide a Cloudinit or Sysprep script to customize this VM.

ADSF path
Start typing for suggestions

Upload a file
Choose File No file chosen

Type or paste script

|
|

FILES TO COPY
Specify external files to copy inside of the guest VM.

Source File ADSF Path Destination Path in VM +

圖. 定制化腳本-輸入選項

Nutanix 提供了幾個指定定制化腳本路徑的選項：

- ADSF Path
 - 使用一個已經上傳到 ADSF 上的檔
- Upload a file
 - 上傳一個準備使用的檔
- Type or paste script
 - CloudInit 腳本或者是 Unattend.xml 文字檔

包含腳本的 CD-ROM 在第一次啟動時，Nutanix 會將使用者資料腳本傳遞給

CloudInit 或者 Sysprep。

輸入格式

平臺支援大量使用者資料登錄格式，以下是可以識別的重要格式之一：

- User-Data Script (CloudInit - Linux)

User-Data Script 是一個簡單 shell 腳本，該腳本會在啟動進程非常靠後階段才會被執行（例如：“rc.local - like”）

該腳本一般使用以 "#!" 開始的 bash script。



下面是 user-data 腳本舉例：

```
#!/bin/bash
touch /tmp/fooTest
mkdir /tmp/barFolder
```

Include File (CloudInit - Linux)

Include File 包含了多組 URL（每個 URL 一行），每個 URL 都會被讀取並且會被執行到其他任何腳本。

該腳本都是以“# include”開始，類似下面的 include 腳本案例：

```
#include
http://s3.amazonaws.com/path/to/script/1
http://s3.amazonaws.com/path/to/script/2
```

Cloud Config Data (CloudInit - Linux)

對 CloudInit 來說 Cloud-config 輸入類型非常典型，該腳本以“#cloud-init”開始，類似下面的格式：

```
#cloud-config

# Set hostname
hostname: foobar

# Add user(s)
users:
  - name: nutanix
    sudo: ['ALL=(ALL) NOPASSWD:ALL']
    ssh-authorized-keys:
      - ssh-rsa: <PUB KEY>
    lock-passwd: false
    passwd: <PASSWORD>

# Automatically update all of the packages
package_upgrade: true
package_reboot_if_required: true

# Install the LAMP stack
packages:
  - httpd
  - mariadb-server
  - php
  - php-pear
  - php-mysql

# Run Commands after execution
runcmd:
  - systemctl enable httpd
```

確認 CloudInit 已執行

CloudInit 運行日誌是保存在 /var/log/ 下的 cloud-init.log 和 cloud-init-output.log

- Unattend.xml (Sysprep - Windows)

Unattend.xml 檔是 Sysprep 在啟動過程中定制化鏡像時用到的內容。該腳本以“<?xml version=”1.0” ?>”開始。

下面顯示了 unattend.xml 文件的案例：

```

<?xml version="1.0" ?>
<unattend xmlns="urn:schemas-microsoft-com:unattend">
    <settings pass="windowsPE">
        <component name="Microsoft-Windows-Setup" publicKeyToken="31bf3856ad364e35"
language="neutral" versionScope="nonSxS" processorArchitecture="x86">
            <WindowsDeploymentServices>
                <Login>
                    <WillShowUI>OnError</WillShowUI>
                    <Credentials>
                        <Username>username</Username>
                        <Domain>Fabrikam.com</Domain>
                        <Password>my_password</Password>
                    </Credentials>
                </Login>
                <ImageSelection>
                    <WillShowUI>OnError</WillShowUI>
                    <InstallImage>
                        <ImageName>Windows Vista with Office</ImageName>
                        <ImageGroup>ImageGroup1</ImageGroup>
                        <Filename>Install.wim</Filename>
                    </InstallImage>
                    <InstallTo>
                        <DiskID>0</DiskID>
                        <PartitionID>1</PartitionID>
                    </InstallTo>
                </ImageSelection>
            </WindowsDeploymentServices>

```

4.4.3 Karbon (容器服務)

Nutanix 提供了使用 **Kubernetes** (當前) 在 Nutanix 平臺上利用持久容器的能力。之前也可以在 Nutanix 平臺上運行 **Docker**，但是由於容器的天生原因無法使用永久資料存儲。

像 **Docker** 這樣的容器技術是不同於硬體虛擬化。傳統的虛擬化，每個虛擬機器都有自己的作業系統 (**OS**) 但是他們共用底層的硬體。而容器他們共用的是底層的作業系統內核，容器中運行的應用和他們之間的依賴關係都是運行在獨立的進程中。

下面的表格對容器和虛擬機器之間做了簡單對比

	虛擬機器	容器
虛擬化類型	基於硬體的虛擬化	基於作業系統內核的虛擬化
量級	重量級	羽量級
提供速度	慢 (秒級或分鐘級)	即時 (毫秒級)
性能	性能有限	超級性能
安全	完全獨立 (安全高)	進程級別獨立 (安全低)

支援配置方法

適用的配置方法如下：

- Hypervisor(s):



- ■ AHV
- 容器：
- ■ Docker 1.13

在 4.7 開始，該解決方案，目前存儲集成只支援 Docker 的容器。但是其他容器 可以在 Nutanix 平臺上作為虛擬機器運行。

容器服務架構

以下組件構成了 Acropolis Container Services

- **Nutanix Docker Machine Driver**：通過 Docker Machine 和 Acropolis Image Service 控制 Docker 的 container host 發佈
- **Nutanix Docker Volume Plugin**：負責與 Acropolis Block Services 交互來創建、掛載，格式化和附加 volumes 到所需的容器時

以下組件構成了 Docker（注意：不是所有都是必需的）

- Docker Image：容器的基礎鏡像
- **Docker Registry**：為 Docker Images 管理空間
- **Docker Hub**：線上容器交易市場 (public Docker Registry)
- **Docker File**：如何創建 Docker 鏡像的 Text file 文本描述檔
- **Docker Container**：運行 Docker 鏡像的實例
- **Docker Engine**：創建、移動、運行 Docker 容器
- **Docker Swarm**：管理 Docker 集群／調度的平臺
- **Docker Daemon**：處理來自 Docker 用戶端的創建、運行和分配容器的請求
- **Docker Store**：可信的企業級容器交易市場

架構

Nutanix 當前支持將 Docker Engine 運行為使用 Docker 而創建出來的虛擬機器裡。這些機器可以是平臺上運行的普通虛擬機器。

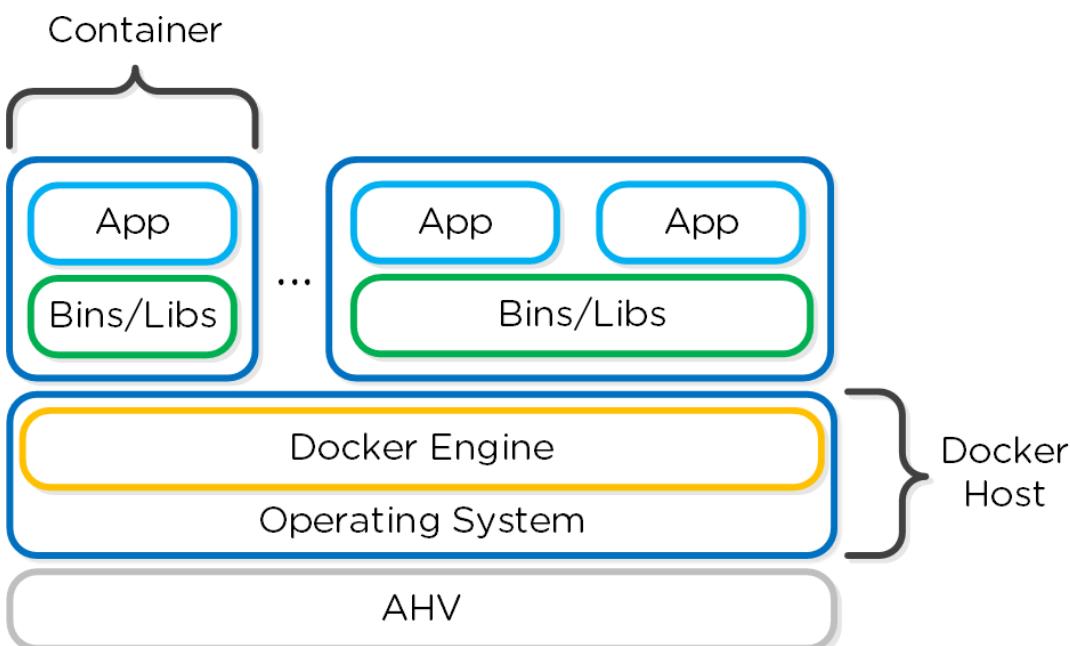


圖.Docker - 高級別架構

Nutanix 開發出 Docker Volume Plugin。利用它，可以通過 Acropolis Block Services 創建、格式化和附加一個 volume 級容器。這樣就可以在容器關機或移動時資料保持不變。

通過 Nutanix Volume Plugin，Acropolis Block Services 在附加 volume 級主機或容器時，資料也能做到永久性。

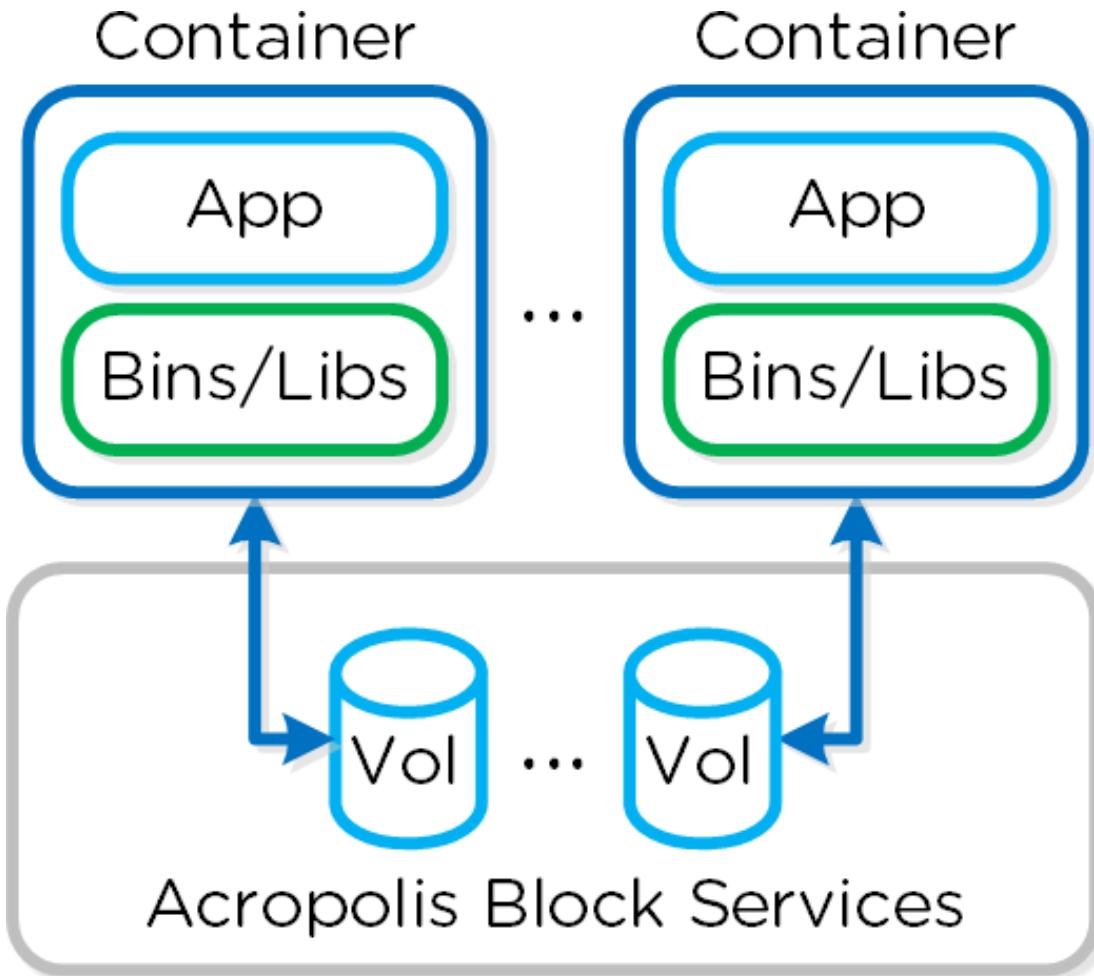


圖. Docker - 塊服務

前提條件

要使用容器服務，需要滿足以下條件：

- Nutanix 集群必須是 AOS 4.7 或以後版本
- Nutanix Docker Host Image 必須下載並且作為一個 Image 保存在 Acropolis 的 Image Service 裡
- Nutanix Data Services IP 必須已經配置好
- Docker Toolbox 必須在 client 機器上安裝且配置
- Nutanix Docker Machine Driver 必須在用戶端的 PATH 裡

創建 Docker Host

當所有的前提條件滿足後，在 Docker 機器上就可以提供 Nutanix Docker Hosts

```
docker-machine -D create -d nutanix \
--nutanix-username <PRISM_USER> --nutanix-password <PRISM_PASSWORD> \
--nutanix-endpoint <CLUSTER_IP>:9440 --nutanix-vm-image <DOCKER_IMAGE_NAME> \
--nutanix-vm-network <NETWORK_NAME> \
--nutanix-vm-cores <NUM_CPU> --nutanix-vm-mem <MEM_MB> \
<DOCKER_HOST_NAME>
```

下圖顯示了創建 Docker 主機的工作流：

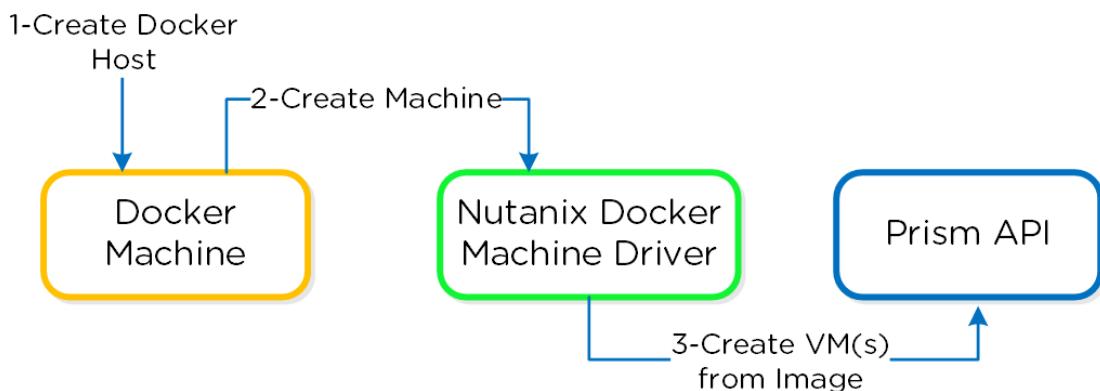


圖. Docker - 主機創建工作流

下一步，在 Docker Machine 上通過 SSH 登錄到最新發佈的 Docker 主機上：

```
docker-machine ssh <DOCKER_HOST_NAME>
```

安裝 Nutanix Docker Volume Plugin :

```
docker plugin install ntnx/nutanix_volume_plugin PRISM_IP= DATASERVICES_IP=
PRISM_PASSWORD= PRISM_USERNAME= DEFAULT_CONTAINER= --alias
nutanix
```

完成後，查看 volume plugin 已經在運行：

```
[root@DOCKER-NTNX-00 ~]# docker plugin ls
ID           Name          Description          Enabled
37fba568078d  nutanix:latest  Nutanix volume plugin for docker  true
```

創建 Docker 容器

當 Nutanix Docker Host 完成部署並且 volume plugin 也啟動，就可以發佈使用永久存儲的容器了。

可以使用典型的 Docker volume 命令架構生成 volume 並制定 Nutanix volume driver。

例如：

```
docker volume create \
<VOLUME_NAME> --driver nutanix
Example:
docker volume create PGDataVol --driver nutanix
```

可以使用標準的 Docker run 命令列並指定 Nutanix volume driver，示例如下：

```
docker run -d --name <CONTAINER_NAME> \
-p <START_PORT:END_PORT> --volume-driver nutanix \
-v <VOL_NAME:VOL_MOUNT_POINT> <DOCKER_IMAGE_NAME>
```

Example:

```
docker run -d --name postgresexample -p 5433:5433 --volume-driver nutanix -v
PGDataVol:/var/lib/postgresql/data postgres:latest
```

下圖顯示了創建容器的工作流：

1-Create container
with volume

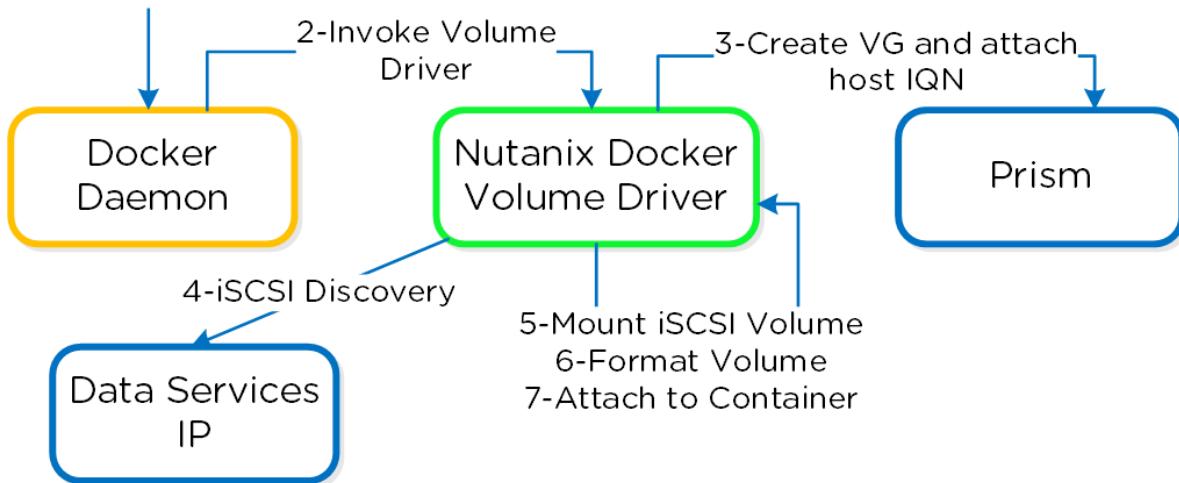


圖.Docker - 容器創建的工作流

現在你就擁有了配備永久存儲的容器！

4.5 備份與容災

Nutanix 提供原生備份和容災（DR）功能，允許用戶將運行在 DSF 上的 VM 和物件備份，還原和災難恢復到本地或雲環境（Xi）。從 AOS 5.11 開始，Nutanix 發佈了一項名為 Leap 的功能，該功能抽象了許多這些概念。有關 Leap 的更多資訊，請參閱“中的“Prism”一章。

下面的章節包含以下內容：

- 實施組件
- 保護對象
- 備份和恢復
- 複製和容災

注意：儘管 Nutanix 提供備份和容災的原生功能選擇，那些借助平臺提供內置特性（VSS、快照等）的傳統備份解決方案（例如 Commvault、Rubrik 等）仍然可以使

4.5.1 實施組件

Nutanix 備份和容災涉及到以下關鍵元件：

保護域（PD／Protection Domain）

- 主要角色：同時保護多個“虛擬機器／檔”的邏輯組
- 描述：一組虛擬機器或檔基於某個相同的保護策略進行複製保護。一個 PD 可以保護一整個容器（Container）或被選中的虛擬機器或文件。

專家提示：

可以針對不同的 RPO／RTO 需求，創建多個不同的 PD。例如可以針對“檔分發”創建專門的 PD，用於 Golden images, ISO 文件的複製等。

一致性組（CG／Consistency Group）

- 主要角色：PD 中多個相關聯的虛擬機器或檔構成的一個子集，以實現故障一致性。
- 描述：PD 中多個相關聯的虛擬機器或檔需要使用故障一致性發起快照。從而確保在虛擬機器或檔回滾時的資料一致性。一個 PD 中可包含多個 CG。

專家提示：

相互依賴的多個應用或服務類虛擬機器（e.g. APP and DB）放置在一個 CG 中，可確保在發生回滾時的資料一致性。

快照計畫（Snapshot Schedule）

- 主要角色：快照和複製計畫
- 描述：為指定 PD 和 CG 中的虛擬機器提供快照、複製的計畫安排

專家提示：

快照計畫應該符合預期 RPO 的要求

保留策略（Retention Policy）

- 主要角色：本地或遠端網站中保留的快照數量
- 描述：保留策略定義了本地或遠端網站中保留的快照數量。注意：在遠端保留/複寫原則配置前，必須先配置遠端網站。

遠程網站 (Remote Site)

專家提示：

保留策略即為虛擬機器或檔的復原點數量

- 主要角色：遠程 Nutanix 集群
- 描述：遠端 Nutanix 集群是作為備份或容災的目標來使用的。

專家提示：

確保目標網站有充足容量（計算／存儲）處理整個網站故障。

某些場景中在單個網站中不同機架之間實施複製／容災也是有意義的。

以下的圖片展示了一個網站內，PD、CG 和虛擬機器/檔間的邏輯關係

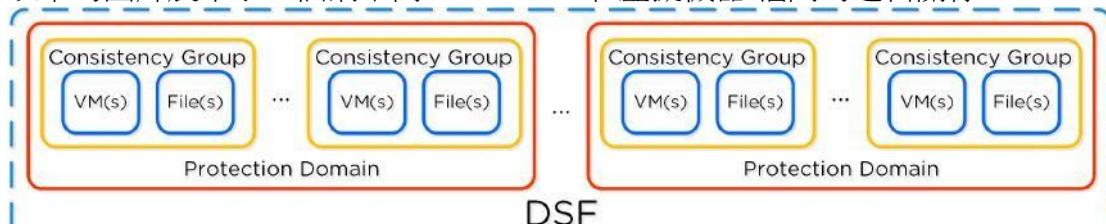


圖.容災組件關係

基於策略的災難恢復和 RunBooks

基於策略的災難恢復和 RunBooks 擴展了基於虛擬機器的災難恢復（PD，CG 等）中定義的功能，並將其抽象為策略驅動的模型。通過關注感興趣的專案（例如 RPO，保留等）並分配給類別而不是直接分配給 VM，可以簡化配置。這也允許適用於所有 VM 的“默認策略”。

注意：這些策略是通過 Prism Central (PC) 配置的。

4.5.2 保護對象

使用如下過程保護物件（VMs、VGs、Files）：

在“Data Protection”頁面，選擇+Protection Domain->Async DR：

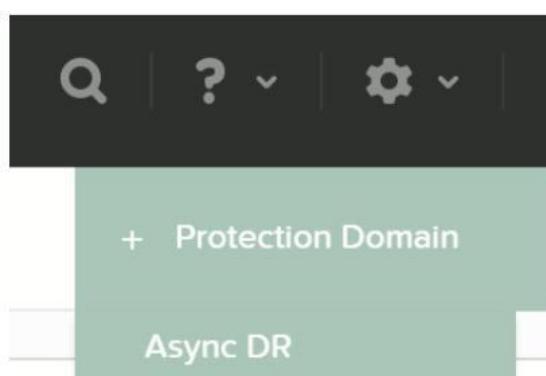


圖. 容災 (DR) – 非同步保護

域指定 PD 名稱並點擊“Create”

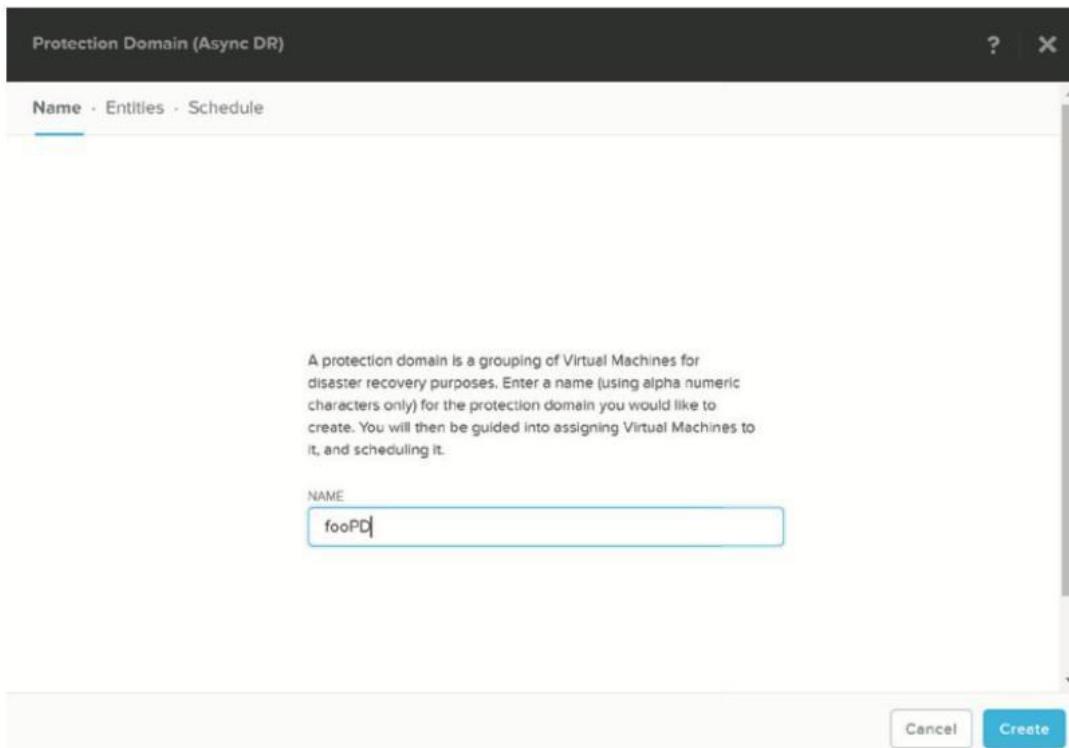


圖. 容災 (DR) – 創建保護域
選擇要保護的物件

Protection Domain (Async DR)

Name · **Entities** · Schedule

Unprotected Entities (1074)

Filter by:

Entity Name

<input type="checkbox"/>	▲ NAME	TYPE
<input checked="" type="checkbox"/>	backup_centosvol	Volume Group
<input checked="" type="checkbox"/>	clonecentos	Volume Group
<input type="checkbox"/>	cloudinit-1	Virtual Machine
<input type="checkbox"/>	FooBar88	Virtual Machine

Pick a Consistency Group ?

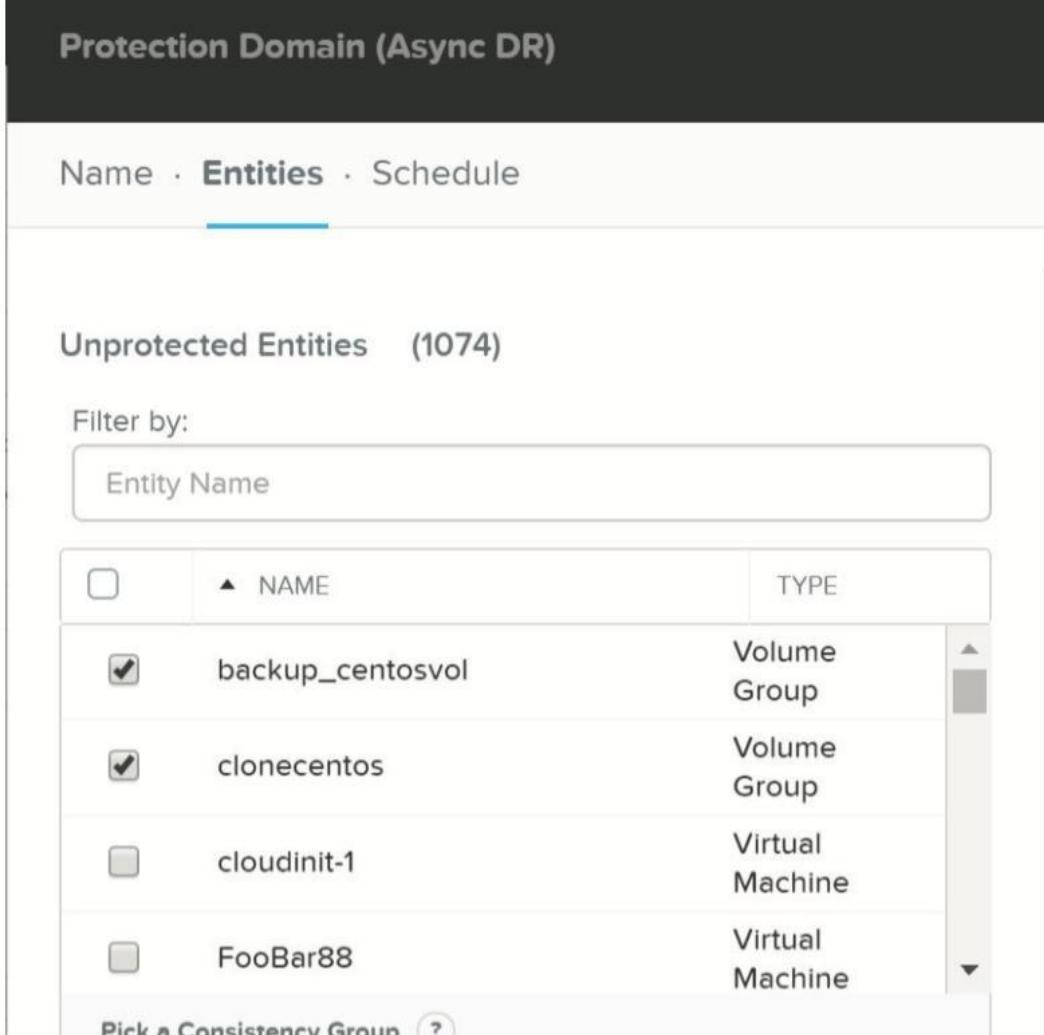


圖. 容災 (DR) – 非同步保護
域點擊“Protect Selected Entities”



Pick a Consistency Group ?

Use Entity Name

Use an existing CG ▾

Create a new CG

Snapshots

Use application consistent snapshots ?

Protect Selected Entities (2) >

圖 . 容災 (DR) – 保護實體
保護物件將顯示在“Protected Entities”中

Protected Entities (2)

Filter by:

Entity Name

CG Name

<input type="checkbox"/>	▲ ENTITY NAME	CG
<input type="checkbox"/>	backup_centosvol	backup_centosvol
<input type="checkbox"/>	clonecentos	clonecentos

圖 . 容災 (DR) – 被保護的實體
點擊“Next”，然後點擊“Next Schedule”來創建快照和複製計畫。
輸入需要的快照頻率、保留策略和複製的遠端網站。

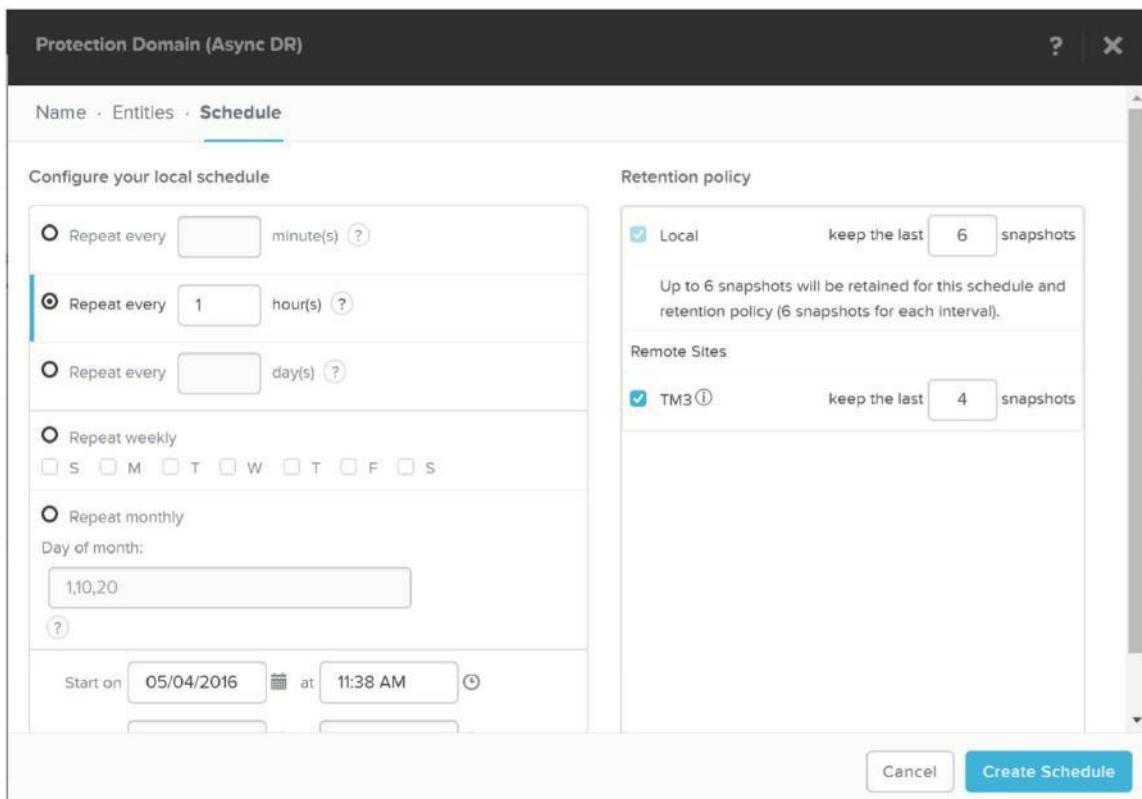


圖 . 容災 (DR) – 創建計畫

點擊“Create Schedule”完成計畫創建。

多保護計畫

可以創建多個快照／複製計畫。例如，一個每小時的本地備份計畫，另一個每天複製到遠端網站的計畫。

需要重點指出的是，Nutanix 不僅可以簡單的將一整個容器（Container）進行保護，還可以提供針對單個虛擬機器或檔級的更加細緻的保護。

4.5.3 備份和恢復

Nutanix 備份功能借助原生的 DSF 快照，由 Cerebro 調用並由 Stargate 執行。這種快照是 0 複製確保有效利用存儲、降低開銷。更多關於 Nutanix 快照的內容在“快照和克隆”章節。

典型備份和恢復操作包括：

- 快照：創建復原點並複製（如果需要的話）
- 恢復：從之前的快照中恢復虛擬機器/檔（替換原有物件）
- 克隆：類似於恢復，但不替換原有物件（使用需要的快照創建新物件）

在“Data Protection”頁，可以看到之前在“Protecting Entities”部分創建的保護域（PD）。

NAME	REMOTE SITES	ENTITY COUNT	NEXT SNAPSHOT TIME	SNAPSHOT EXCLUSIVE USAGE	B/W USED (Tx)	B/W USED (Rx)	ONGOING	PENDING
fooPD	TM3	2	05/04/2016, 03:38:00 PM	-	0 KBps	0 KBps	0	0

圖. 容災 (DR) – 查看保護域
一旦選定目標 PD，可以看到不同選項：

Take Snapshot Migrate Update Delete

圖.容災 (DR) – 保護域動作

如果點擊“Take Snapshot”，可以為選擇的 PD 臨時創建快照並按需要複製到遠端網站：

Replicate Protection Domain ? X

Select one or more targets to replicate to. This is a one time replication that can start now or at a later time.

LOCAL
REMOTE SITES
 TM3

REPLICATION START TIME

Now

RETENTION TIME

No Expiration

Create application consistent snapshot

Cancel Save

圖. 容災 (DR) – 創建快照
“Migrate” PD 會將對象容錯移轉到遠端網站：

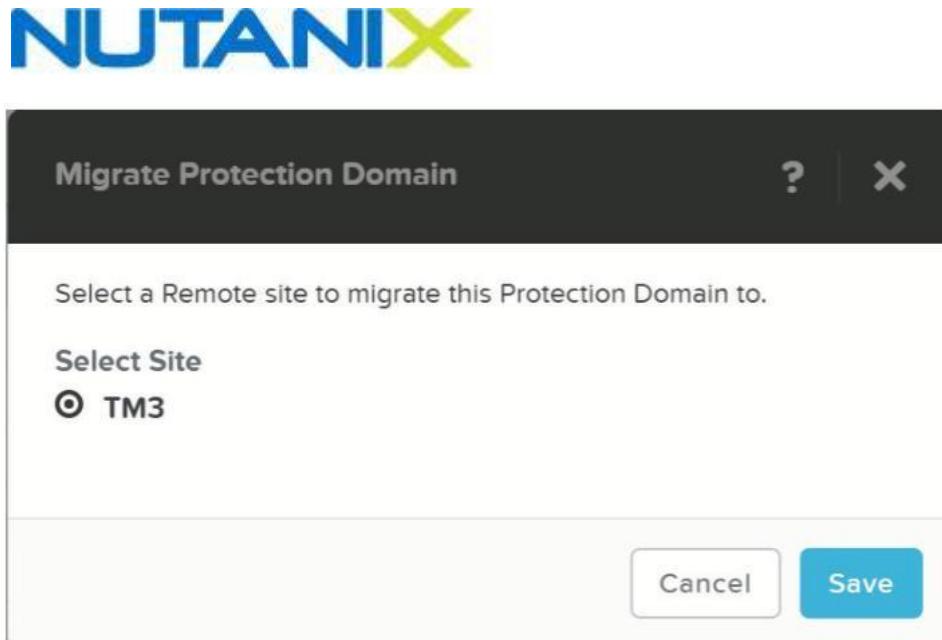


圖. 容災 (DR) - 遷移

在遷移（受控容錯移轉）的情況下，系統將創建一個新快照，進行複製，然後使用新創建的快照升級另一個網站。

專家提示

在 AOS 5.0 及更高版本中，您可以利用單節點複製達到資料保護。

在下面表格中查看 PD 快照資訊：

Replications	Entities	Schedules	Local Snapshot	Remote Snapshots	Metrics	Alerts	Events
				<input type="checkbox"/> Include Expired · 4 Snapshots · < > · ·	search in table		
<input type="checkbox"/>	ID		CREATE TIME	RECLAIMABLE SPACE	EXPIRY TIME	VM RECOVERY	
<input type="checkbox"/>	54404		05/04/2016, 02:38:00 PM	Processing	05/04/2016, 08:38:00 PM	Recovery Details	Details · Restore ·
<input type="checkbox"/>	54357		05/04/2016, 01:38:00 PM	Processing	05/04/2016, 07:38:00 PM	Recovery Details	Details · Restore ·
<input type="checkbox"/>	54310		05/04/2016, 12:38:00 PM	Processing	05/04/2016, 06:38:00 PM	Recovery Details	Details · Restore ·
<input type="checkbox"/>	54261		05/04/2016, 11:39:18 AM	0	05/04/2016, 05:39:18 PM	Recovery Details	Details · Restore ·

圖. 容災 (DR) – 保護域動作

從這裡可以恢復或者克隆 PD 的快照：

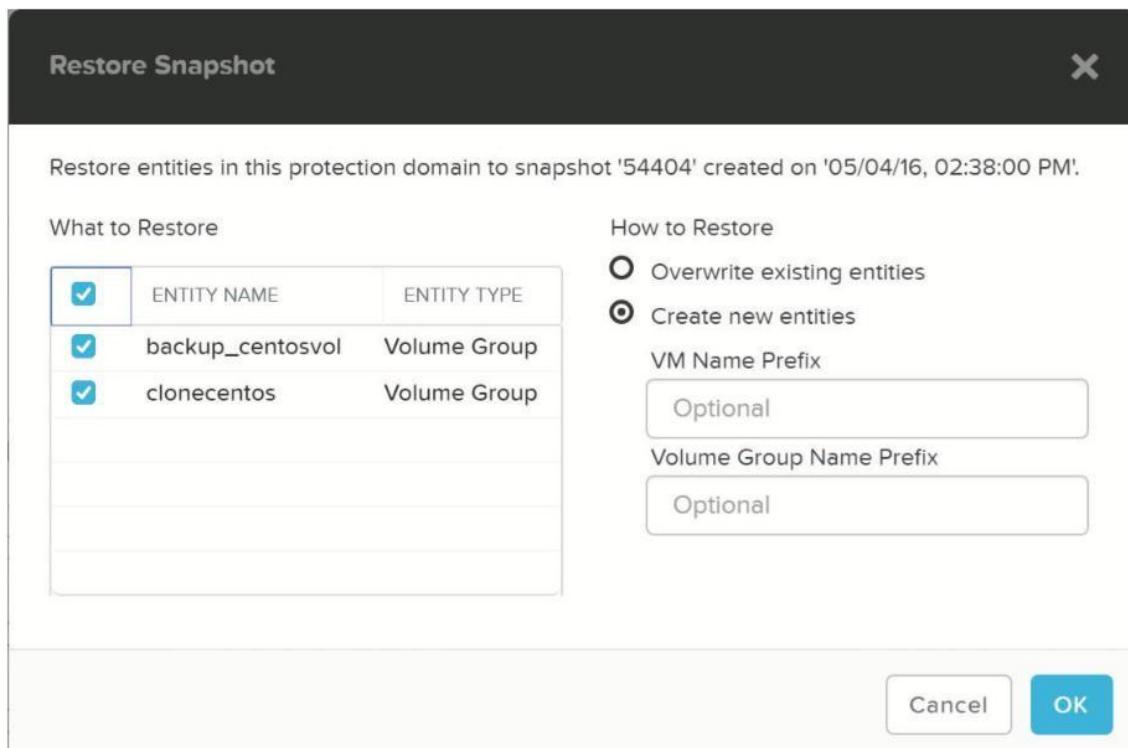


圖. 容災 (DR) – 恢復快照

選擇“Create new entities”類似於克隆 PD 快照創建指定首碼的新物件。另外，“Overwrite existing entities”會使用快照時間點上的物件替換現有物件。

存儲型備份目標設備

僅僅是用於備份／歸檔目的，可以在遠端網站配置存儲型 Nutanix 集群用作備份目標。這樣資料可以複製到存儲型集群或者從存儲型集群進行複製。

4.5.4 應用一致性快照

Nutanix 提供原生的 VmQueisced Snapshot Service (VSS) 功能靜默作業系統和應用操作，以確保完成應用一致性快照。

支援的配置

VmQueisced Snapshot Service (VSS)

VSS 是典型的用於 Windows 的名詞，指 Volume Shadow Copy Service。但是因為該方案既用於 Windows，也用於 Linux，所以 Nutanix 將該名詞修改為 VmQueisced Snapshot Service。

該方案用於 Windows 和 Linux 客戶虛擬機器，包括以下版本（列表可能不全，參考文檔獲得完整支持列表）：

- Hypervisor :
 - ESX
 - AHV
- Windows :
 - 2008R2, 2012, 2012R2
- Linux :
 - Centos 6.5/7.0
 - RHEL 6.5/7.0
 - OEL 6.5/7.0
 - Ubuntu 14.04+
 - SLES11SP3+

前提條件

使用 Nutanix VSS 快照的必要條件如下：

- Nutanix 平臺
 - 必須配置集群 Virtual IP (VIP)
- 客戶作業系統／使用者虛擬機器
 - 必須安裝 NGT
 - 必須能夠連接集群 VIP 的 2074 埠
- 容災配置
 - 客戶虛擬機器所在的 PD 必須啟用“Use application consistentsnapshots”

架構

4.6 版本開始使用原生的作為 Nutanix 客戶工具包的一部分的 Nutanix Hardware VSS provider 來實現應用一致性。更多內容在“Nutanix Guest Tools”章節。

以下圖片顯示 VSS 架構的高層視圖：

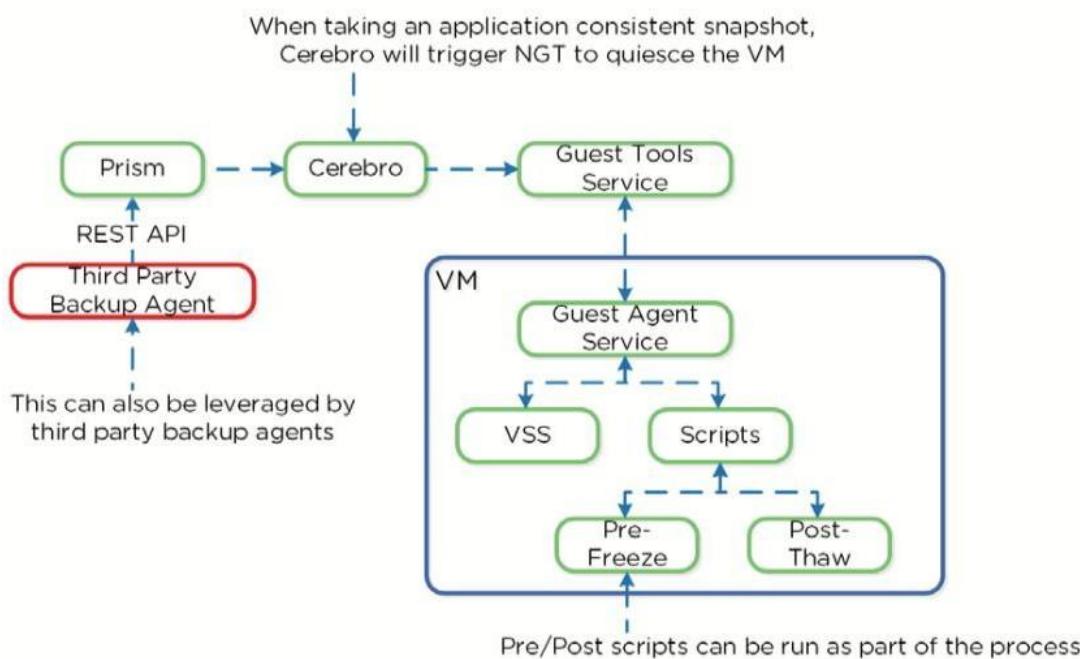


圖. Nutanix VSS – 高層視圖

當保護虛擬機器時，可以通過下面的普通資料保護流程並選擇“Use applicationconsistent snapshots”執行應用一致性快照。

啟用／禁用 Nutanix VSS

為用戶虛擬機器啟用 NGT 時，Nutanix VSS 快照功能默認啟用。但是需要使用以下命令關閉該功能：

```
ncli ngt disable-applications application-names=vss_snapshot vm_id=<VM_ID>
```

Windows VSS 架構

Nutanix VSS 方案集成在 Windows VSS 框架中。下圖顯示了該架構的高層視圖：

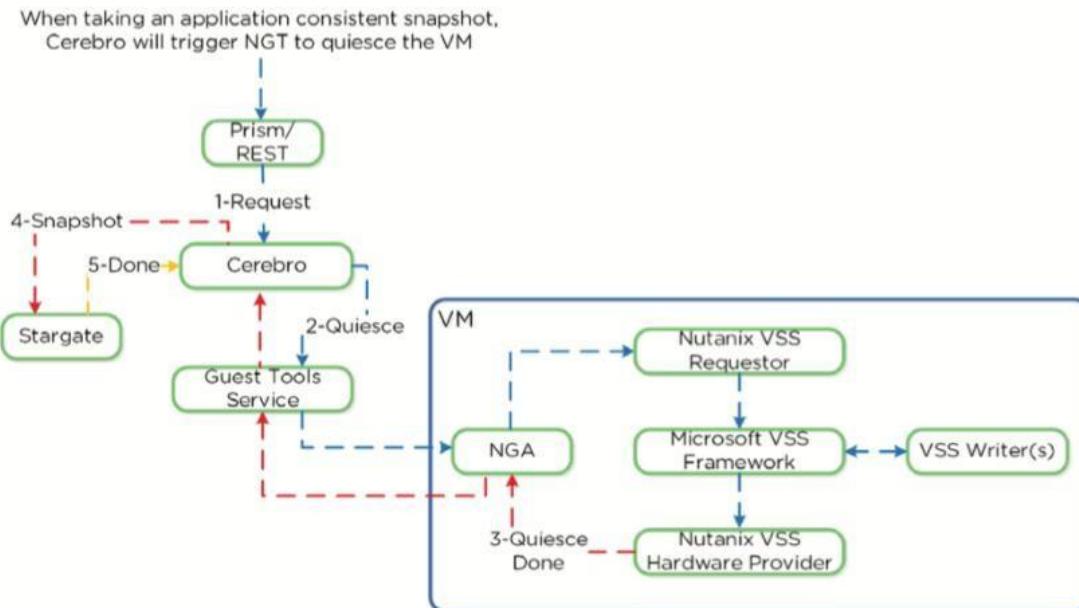


圖 . Nutanix VSS – Windows 架構

一旦 NGT 安裝，就可以看到 NGT 代理和 VSS Hardware Provider Service :

SERVICES					
Filtered results 2 of 135 total					
Server Name	Display Name	Service Name	Status	Start Type	
WIN-7M16SPEHU1L	Nutanix VSS Hardware Provider	Nutanix VSS Hardware Provider	Running	Automatic	
WIN-7M16SPEHU1L	Nutanix Guest Tools Agent	Nutanix Guest Agent	Running	Automatic	

圖 . VSS 硬體供應商

Linux VSS 架構

Linux 方案類似於 Windows 方案，但是借助於腳本而不是 Microsoft VSS 框架實現，因為 Microsoft VSS 框架在 Linux 發行版本中不存在。

Linux VSS 架構高層視圖如下所示：

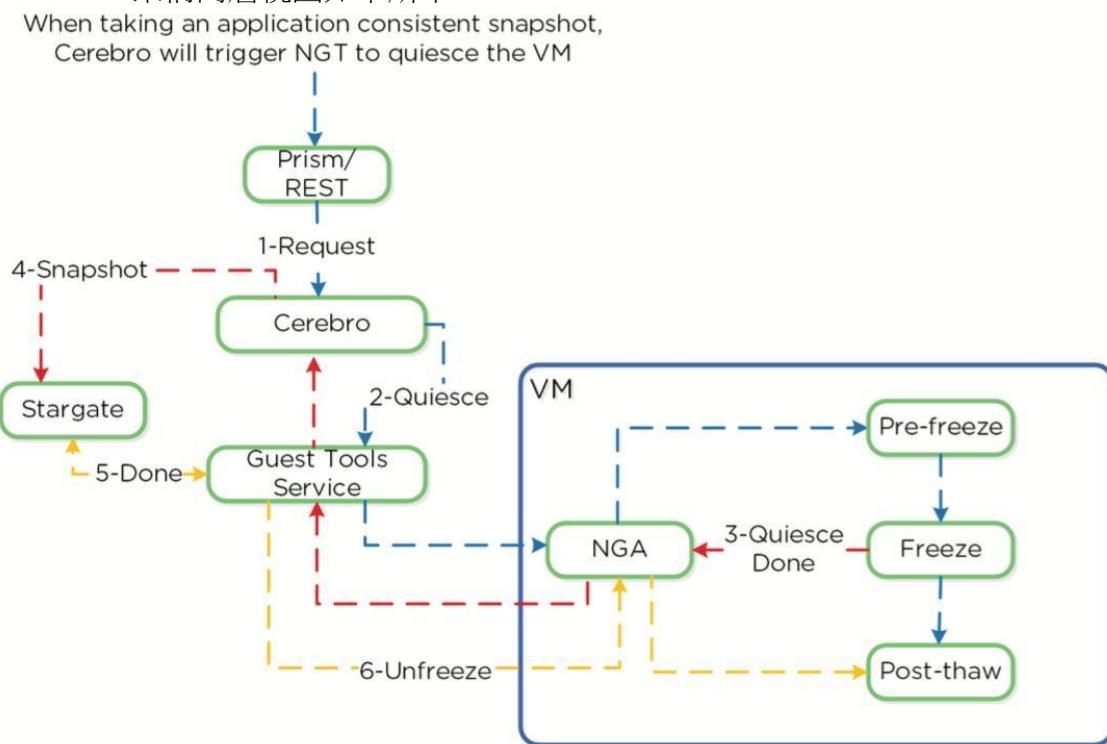


圖 . Nutanix VSS – Linux 架構

pre-freeze 和 post-thaw 腳本位於以下目錄：

- Pre-freeze : /sbin/pre-freeze
- Post-thaw : /sbin/post-thaw

消除 ESXi 震盪 (Stun)

ESXi 使用 VMware 用戶端工具提供原生應用一致性快照。但是在這個過程中，會創建差異磁片來處理新的寫 I/O，ESXi 為了重新映射虛擬磁片到新的差異檔，會“震盪”虛擬機。VMware 快照刪除時也會產生震盪。

在震盪過程中，虛擬機器本身的作業系統不能執行任何操作，基本處於停滯狀態（例如 pings 會失敗，沒有 I/O）。震盪時間依賴於 vmdk 的數量和 datastore 中繼資料操作的處理速度（例如創建新的差異磁片等）

使用 Nutanix VSS 完全規避了 VMware 快照／震盪過程，對性能或者虛擬機器／作業系統可用性幾乎無影響。

4.5.5 複製和容災 (DR)

Nutanix 提供原生的容災 (DR) 和複製功能，它們構建於“快照”&“克隆”等功能之上。Cerebro 是在分散式存儲 (DSF) 中負責管理容災和複製的組件。Cerebro 運行於每個節點之中，通過內部選舉產生 Master (類似於 NFS Master)，並由此節點管理複製任務。如果當 Cerebro Master 所在的 CVM 發生故障，剩餘的節點將選舉出新的 Master。通過<CVM IP>:2020，即可打開 Cerebro 的相關介面。容災功能可以分解為以下關鍵要點：

- 複寫拓撲
- 複製週期
- 全域去重

複寫拓撲

一直以來，有幾種主要的複製網路拓撲：點對點 (Site to site)，菊輪鍊 (hub and spoke)，全網狀／部分網狀 (full and/or partial mesh)。相對於傳統的方案，它們僅提供“點到點”或“菊輪鍊”的方式，Nutanix 提供“全網狀”或更加靈活的“多到多”的拓撲方式。

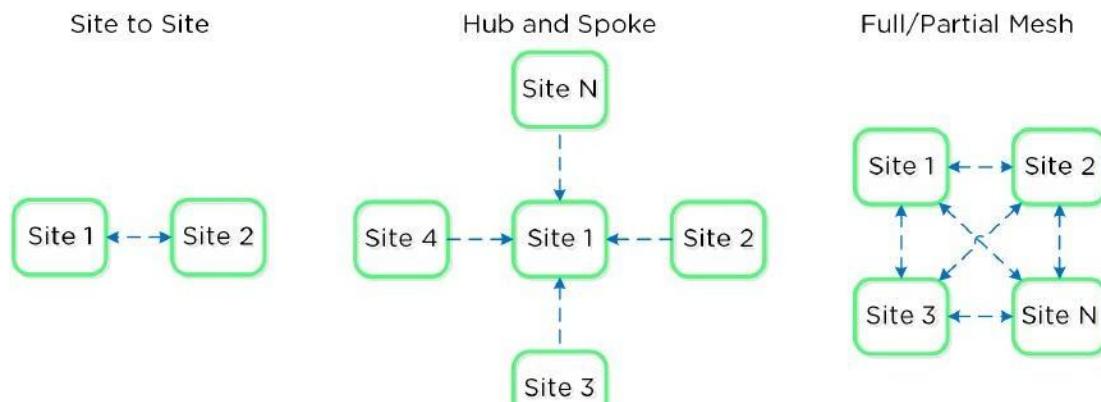


圖 . 複寫拓撲結構示例

這將讓管理員能夠靈活的配置複製功能，從而更好的滿足公司需求。

複製週期

Nutanix 通過 Cerebro 實現資料的複製。Cerebro 服務分為 Cerebro Master 和 Cerebro Slave。Cerebro Master 由動態選舉產生，除 Cerebro Master 節點之外的 CVM 中，均運行 Cerebro 從屬服務 (Cerebro Slaves)。一旦“Cerebro Master”所對應的 CVM 啟機，新的 Master 將被自動選舉產生。

Cerebro Master 負責委派任務給本地的 Cerebro 從屬節點，以及協調遠端的 Cerebro Master，實現容災資料複製。

在複製過程中，Cerebro Master 將負責確認哪些資料需要被複製，同時將任務委派給 Cerebro 從屬節點，隨後將告知 Stargate 哪些資料需要被複製，需要被複製到哪裡。

在複製過程中，複製資料在多個層面被保護。源端 Extent 讀使用校驗碼保證來源資料的一致性（類似於 DFS 讀），在目標端，新的 Extent(s)也會生成校驗碼（類似於DFS 寫）。TCP 提供網路層面的一致性。

以下是相關架構的示意圖：

專家提示：

當使用代理方式配置遠端網站時，一般都使用集群 IP。此 IP 將宿主在 Prism 主節點（Prism Leader）上。即使任一 CVM 容機，也可保證該 IP 可用。

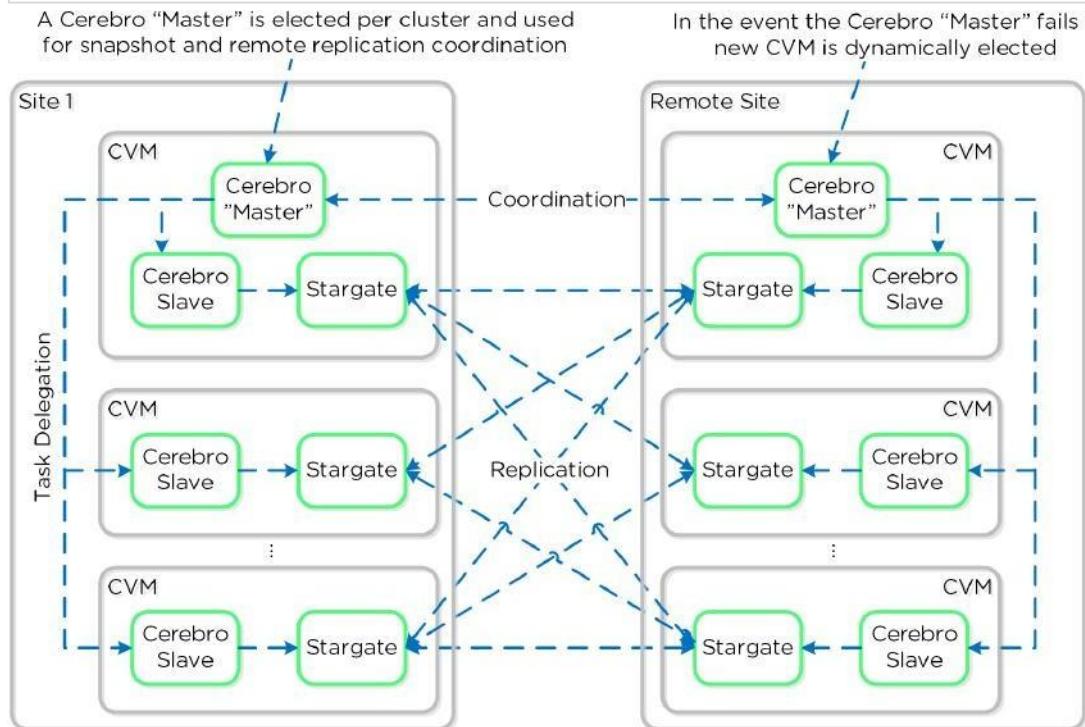


圖 . 複製架構

我們也可以通過配置代理，使用橋接的方式承載整個集群的協調和複製的資料流程。

專家提示：

當配置代理伺服器來連接到遠端網站時，請一直使用集群 IP 位址（由 Prism Leader 角色提供並一直可用，甚至在 CVM 當機時仍然有效）

以下是使用代理的架構示意圖：

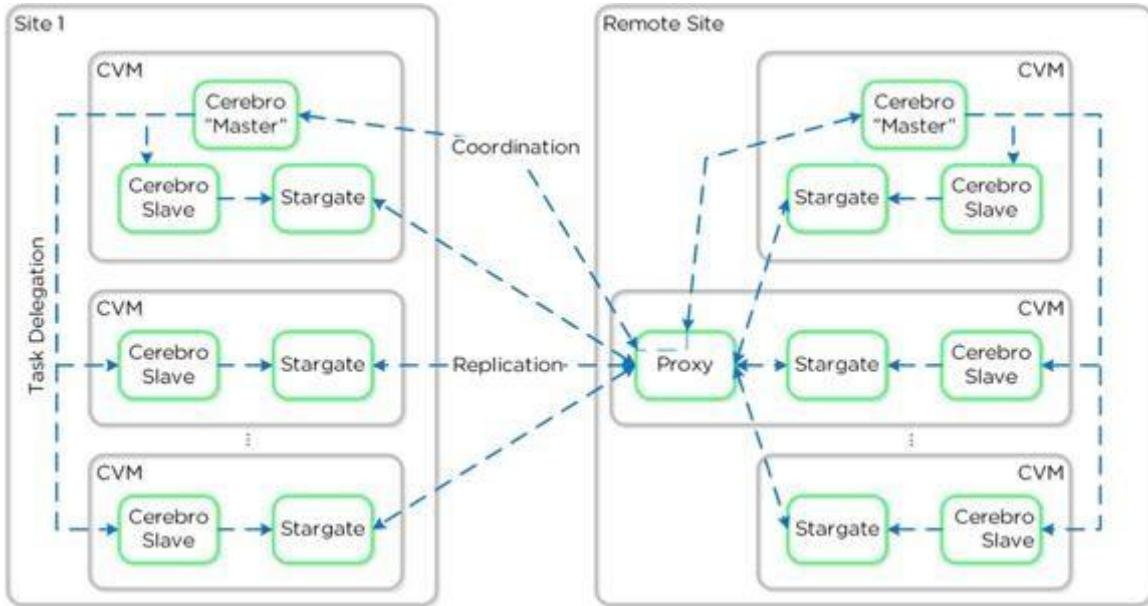


圖 . 複製架構 - 代理 (Proxy)

在某些情況下，我們也可以配置 SSH 隧道連結兩個 CVM。

專家提示：

這只適用於非生產環境下，並請使用集群 IP 以確保高可用。

以下是使用 SSH 隧道的架構示意圖：

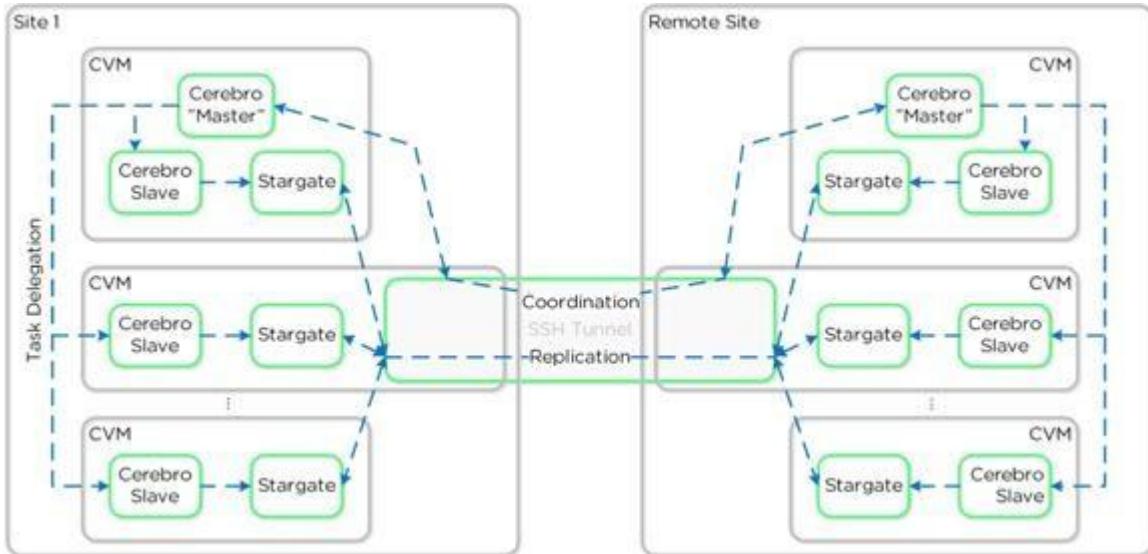


圖 . 複製架構 - SSH 隧道

全域去重

正如之前“Elastic Deduplication Engine”章節中所提到的，分散式存儲 (DSF) 可以僅通過更新中繼資料 (Metadata) 指標實現重複資料刪除。同樣的實現方式也可被用

于容災和複製中。通過網路發送資料之前，DFS 將檢查遠端網站中是否已經存在相關資料的記錄(fingerprint)。如果有，則不發送資料，僅更新中繼資料。如果遠端網站中沒有此資料記錄，資料將被壓縮並發送至目標網站中。此時，該資料將被用於兩個網站中的重複資料刪除。

以下是 3 個網站的部署示例，每個網站包含多個 PD：

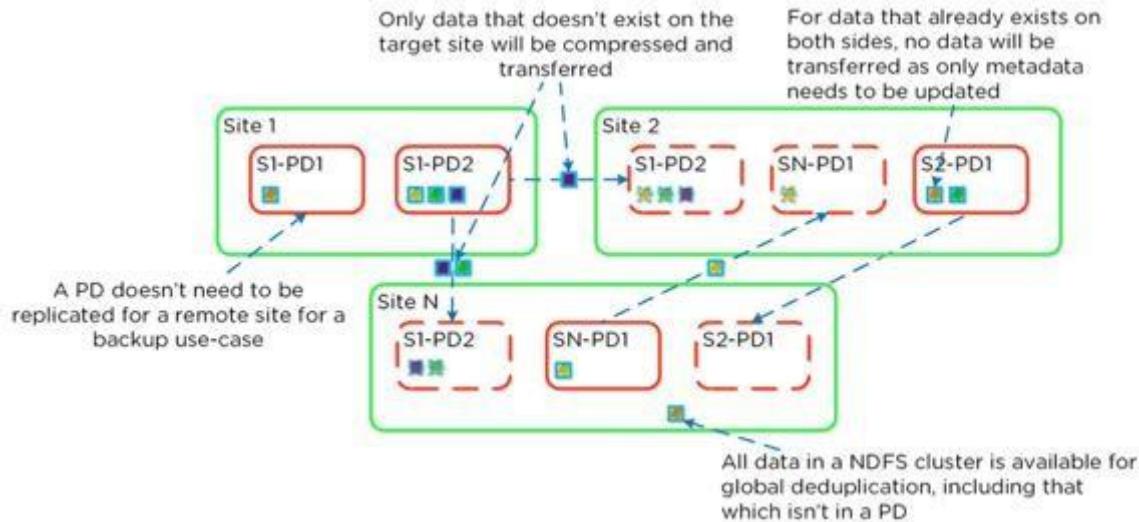


圖 . 複製消重

注意：

Fingerprinting 功能必須在源端和目標端的 Container／vstore 中都開啟，實現複製間的重複資料刪除。

4.5.6 近同步複製技術 (NearSync)

基於前面提到的傳統非同步複製功能；Nutanix 在新版本中引入了近同步複製技術 (NearSync) 的支持。

NearSync 提供了兩全其美的功能：除了非常低的 RPO (與同步複製 (metro) 相近) 之外，對主 I/O 延遲 (與非同步複製相同) 的影響為零。這意味著允許用戶具有非常低的 RPO，而又不會產生對寫入的同步複製的開銷。

此功能使用稱為羽量級快照 (LWS) 的新快照技術。與使用非同步方式的傳統基於虛擬磁片的快照不同，它利用標記並且完全基於 OpLog 來完成 (與在 Extent Store 中完成的虛擬磁片快照相比)。

Mesos 是一項新增的服務，用於管理快照層並抽象完整/增量快照的複雜性。
Cerebro 繼續管理高層架構和策略（例如一致性組等），而 **Mesos** 負責與 **Stargate** 交互並控制 **LWS** 生命週期。

下圖顯示了 **NearSync** 元件之間的通信示意例：

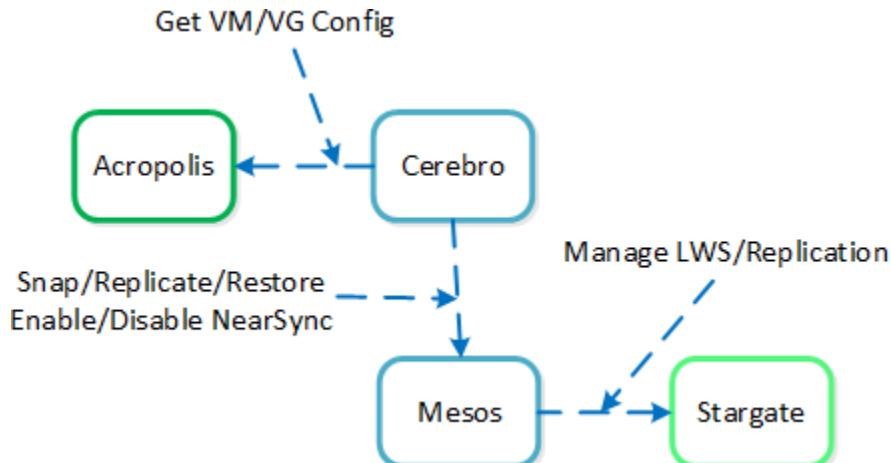


圖. NearSync 組件交互

當使用者配置快照頻率 ≤ 15 分鐘時，將自動利用 **NearSync** 技術進行保護。在此之後，初始種子快照被執行然後被複製到遠端網站。一旦這個過程在 60 分鐘內完成

（可以是第一個或第 n 個），除了開始 **LWS** 快照複製之外，另一個種子快照將被立即執行和複製。一旦第二個種子快照完成複製，所有已複製的 **LWS** 快照將變為有效，並且系統處於穩定的 **NearSync** 運行狀態中。

下圖顯示了啟用 **NearSync** 執行的示例時間表：

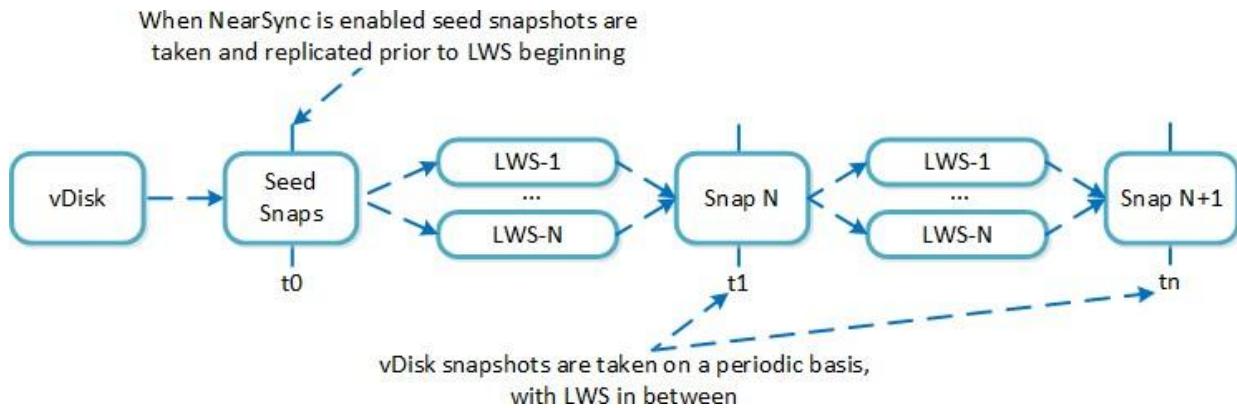


圖. NearSync 複製生命週期

在穩定的運行狀態下，每隔一小時就會執行一次虛擬磁片快照。除了 **LWS** 之外，遠端網站不是將快照發送到遠端網站，而是基於之前的虛擬磁片快照和 **LWS** 來組成虛擬磁片快照。

如果 NearSync 不同步（例如網路中斷，廣域網路延遲等原因）導致 LWS 複製耗時大於 60 分鐘，系統將自動切換回基於虛擬磁片的快照。當其中一個快照複製在 60 分鐘內完成，系統將立即執行另一個快照，同時開始複製 LWS。在完整快照完成後，LWS 快照將變為有效，系統處於穩定的 NearSync 狀態中。此過程與初次啟用 NearSync 的情況類似。

當基於 LWS 的快照恢復（或克隆）時，系統將克隆最新的虛擬磁片快照並逐漸應用 LWS，直到達到所需的 LWS。

下圖顯示了如何恢復基於 LWS 的快照的示例：

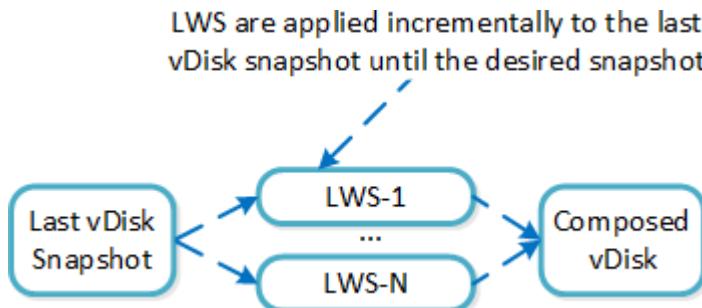


圖. 從 LWS 恢復 vDisk

4.5.7 城域高可用性 Metro Availability

Nutanix 具有“Stretch Clustering”的能力，允許它的計算和存儲集群可以跨越多個物理網站。這種部署架構下，計算集群可以跨越兩個網站並共用一個存儲池。

這將 VM HA 集群從一個網站擴展到兩個網站之間，提供近乎等於 0 的 RTO。這種部署架構下，每個網站都有自己的 Nutanix 集群，但其中容器（Container）的寫操作被同步複製到遠端的網站之中。

以下是此架構的詳細示意圖：

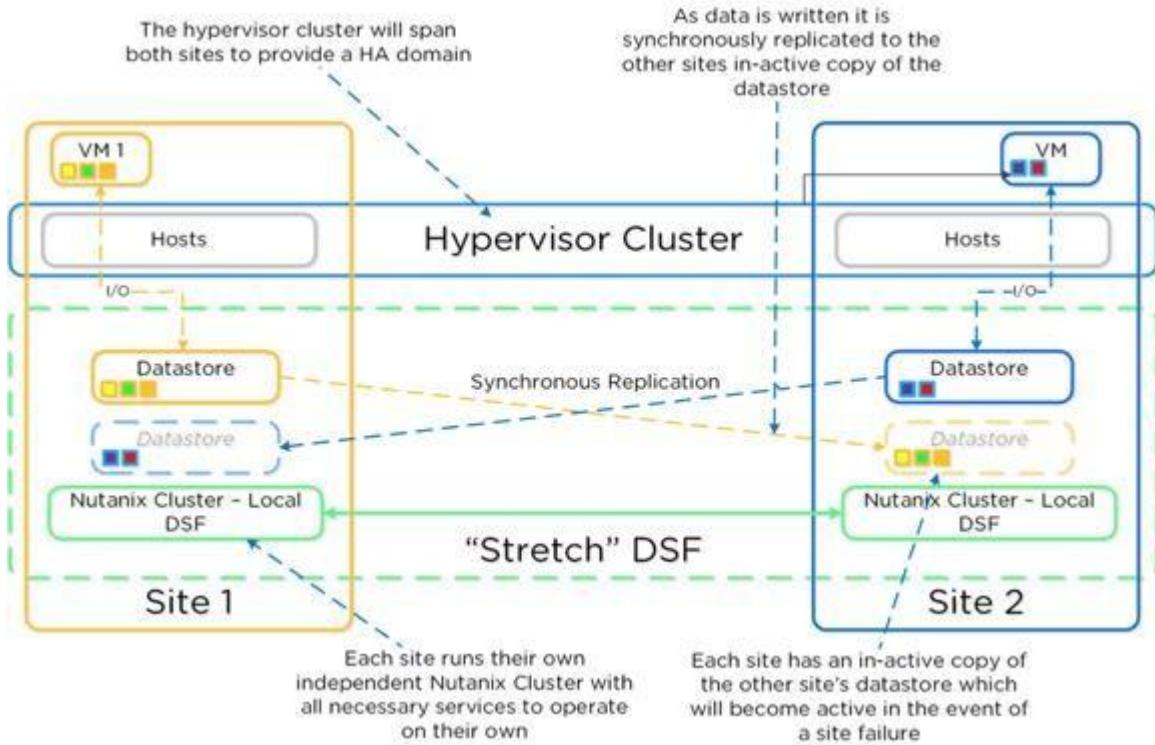


圖. 城域高可用性 - 正常狀態

如果網站發生故障，則會發生 **HA** 事件，在該事件中可以在另一個網站上重新啟動 VM。容錯移轉過程通常是手動過程。在 AOS 5.0 版本中，可以配置 **Metro Witness**，它可以自動執行容錯移轉。見證人可以通過門戶網站下載，並可以通過 Prism 進行配置。

以下是一個網站發生故障的示意圖：

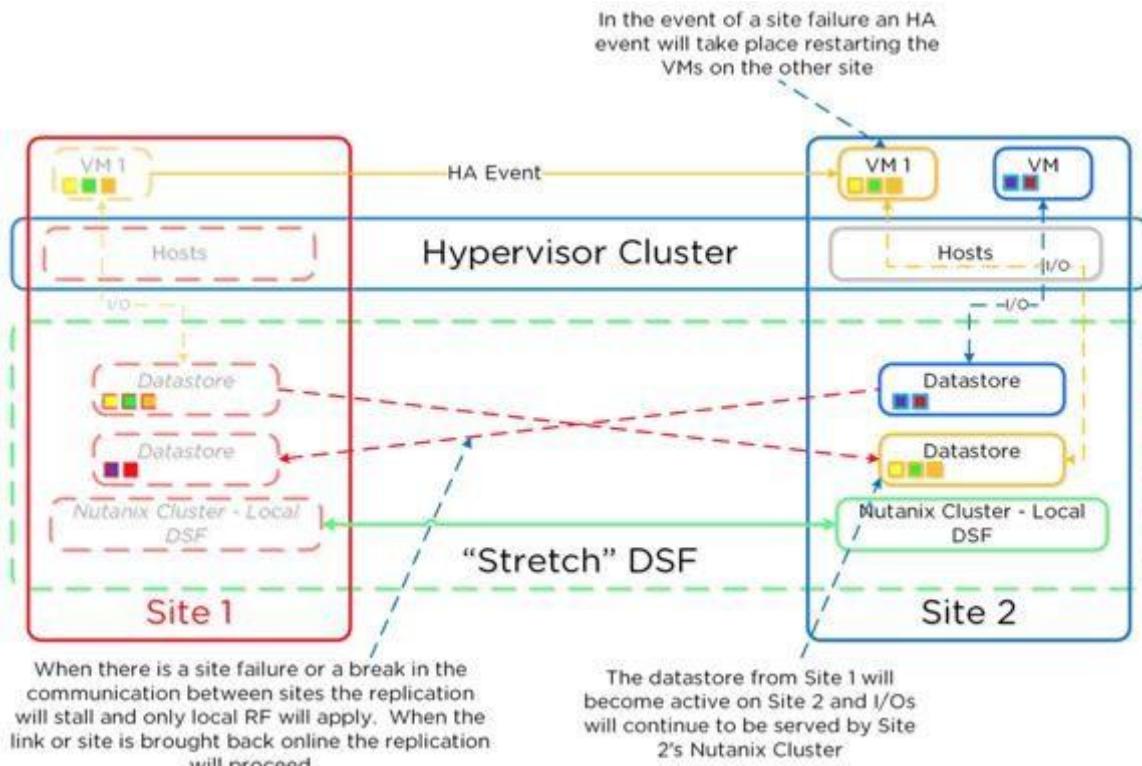


圖 . 城域高可用性 - 網站故障

如果網站間的網路鏈路故障，集群將分別獨立運行。一旦網路鏈路修復，網站將進行增量同步，並重新恢復資料同步。

以下是網路鏈路故障的示意圖：

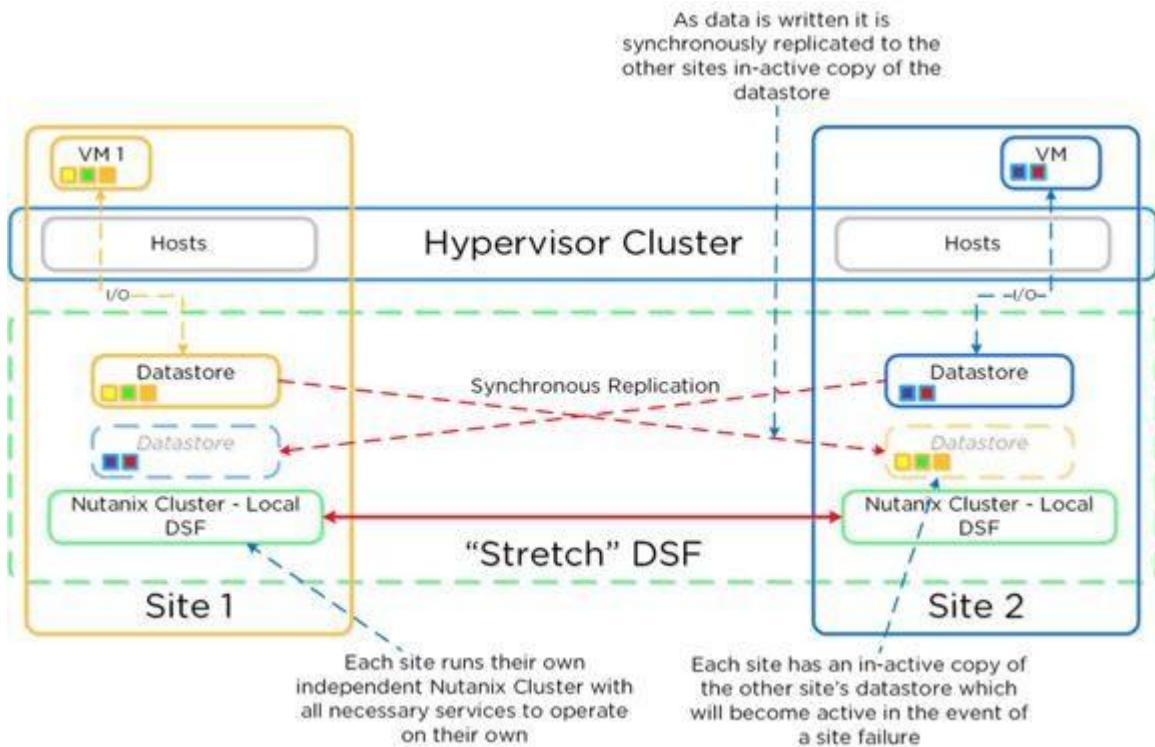


圖 . 城域高可用性 - 連接故障

4.5.8 雲連結

雲連結 (Cloud Connect) 功能可以很好的擴展分散式存儲原生的容災／複製功能（當前支持 Amazon Web Services, 或 AWS）。注意：該功能當前僅適用於備份／複製。

該過程非常類似于原生的容災／複製功能去創建一個遠端網站，實際上是一個“雲端網站”被創建出來。當此雲端網站被創建時，Nutanix 將自動在 EC2 (當前為 m1.xlarge) 或 Azure 虛擬機器 (當前為 D3) 創建一個單節點的集群作為目標端。

運行在 AWS 上的雲實例也是基於跟本地集群相同的 NOS 原始程式碼構建。這就意味著所有原生的複製功能 (例如：全域重複資料刪除、兩地三中心等) 可以直接使用。在這個示例中，有多個 Nutanix 集群使用了雲連結，它們可以共用區域裡同一個實例，或配置一個全新的實例。

雲實例的存儲是使用由 S3(AWS)或者 BlobStore(Azure)提供的邏輯磁片，稱為“雲磁片”。資料存儲為通常的 Egroup，它在物件存儲中作為檔存放。

以下是基於 AWS 的雲端網站雲連結的示意圖：



Cloud (AWS) of Virtual Machine

圖. 雲連結 - 地區

由於基於 AWS 創建的雲端網站類似于 Nutanix 的普通遠端網站，集群可將資料複製到多個不同地域，從而獲得更高的業務連續性（例如，地域性的大規模停電，仍可保證資料可用）：

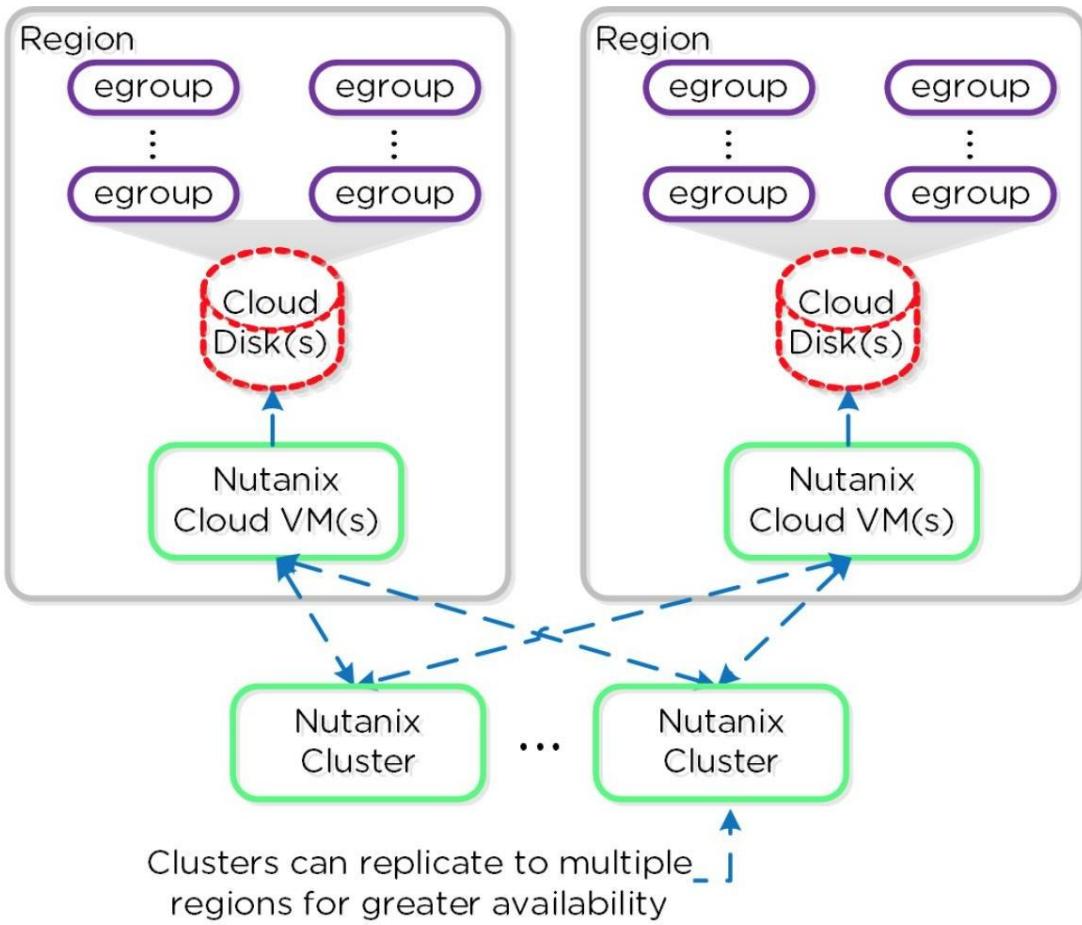


圖 . 雲連結 - 多地區

同樣的複製／保留策略也可被用於雲連結的資料複製之中。當資料或快照失效或過期時，雲集群將執行必要的清理動作。

如果複製的頻率不高（每天一次或每週一次），甚至可以配置為在複製開始前才啟動雲實例，並在複製完成後，將其關機。

複製到雲區域中的資料也可隨時下傳、恢復到現有的或新建的 Nutanix 集群中，當然該集群需要配置、連結到雲端網站。

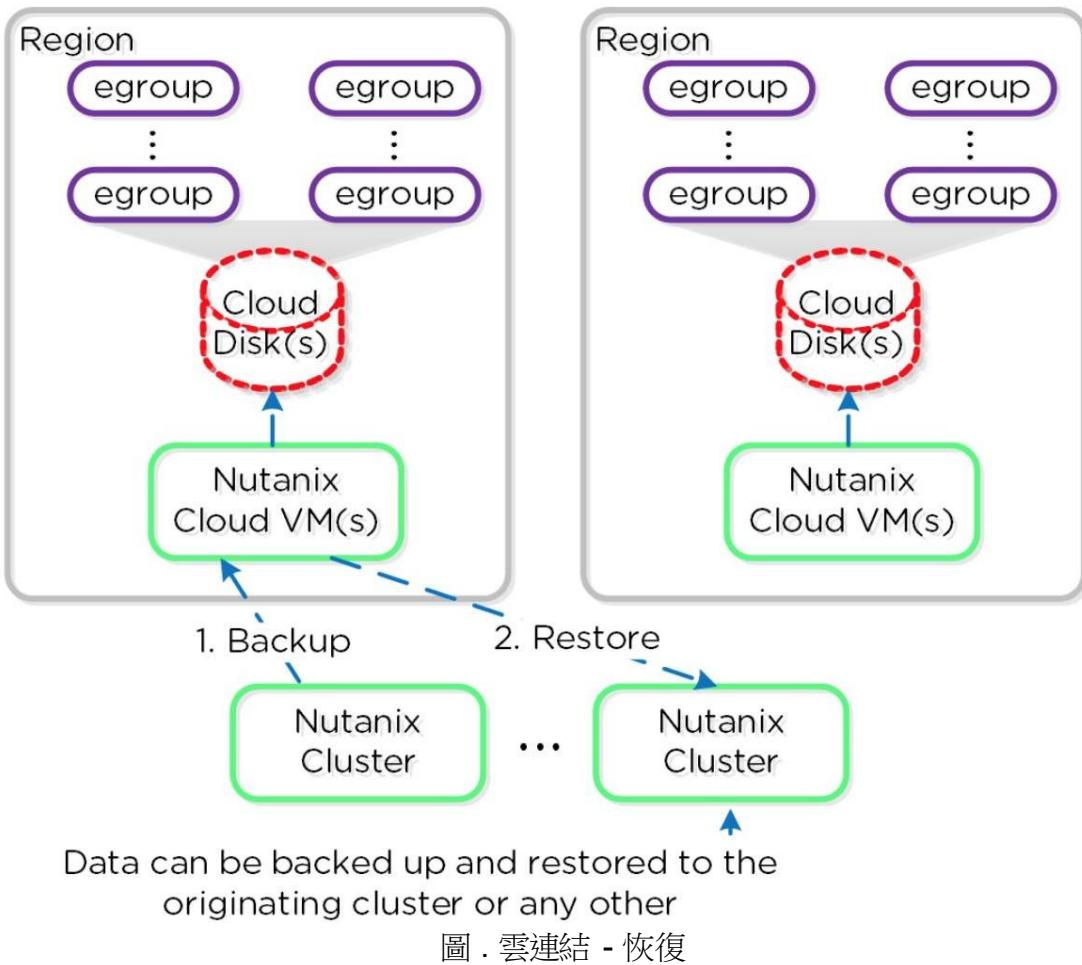


圖 . 雲連結 - 恢復

4.6 Application Mobility Fabric*

4.7 管理

4.7.1 重要頁

除了標準管理頁面之外，高級頁用來監視詳細統計資訊和指標，URL 訪問方式如下：<http://<Nutanix CVM IP/DNS>:<Port/path>> (下文中提到的)。示例：
<http://MyCVM-A:2009>，注：如果你在與 CVM 不同的子網訪問頁面，需要禁用 CVM 防火牆。



這是 Stargate 頁用於監控後端存儲，只能由高級使用者使用。我將發表一篇帖子，介紹 2009 年的頁面和需要尋找的內容。

2009/latency Page

這是 Stargate 頁用於監控後端延遲。

2009/vdisk_stats Page

這是 Stargate 頁用於顯示各種 vDisk 統計資訊，包含 I/O、延遲、寫命中(e.g., OpLog, eStore)、讀命中(cache, SSD, HDD, etc.)等柱狀圖。

2009/h/traces Page

這是 Staegate 頁用於監控操作的活躍跟蹤

2009/h/vars Page

這是 Stargate 頁用於監控各種計數器

2010 Page

這是 Curator 頁用於監控 Curator 運行

2010/master/control Page

這是 Curator 控制頁用於手工啟動 Curator 工作

2011 Page

這是 Chronos 頁監控通過 Curator 監控工作和任務計畫

2020 Page

這是 Cerebro 頁用於監控保護域、複製狀態和 DR

2020/h/traces Page

這是 Cerebro 頁用於監控 PD 操作和複製的實施追蹤

2030 Page



這是 Acropolis 主要頁，顯示主機環境詳細資訊、正在運行的任務和網路詳細資訊。

2030/sched Page

這是一個 Acropolis 頁面，通過顯示虛擬機器資源調度資訊說明決定虛擬機器運行位置，這頁顯示虛擬機器運行的主機的可用資源。

2030/tasks Page

這是一個 Acropolis 頁面用於顯示 Acropolis 任務和狀態，你可以點擊任務 UUID 來獲得任務的詳細 JSON 資訊。

2030/vms Page

這是一個 Acropolis 頁用於顯示 Acropolis 虛擬機器及虛擬機器的詳細資訊，你可以點擊虛擬機器名稱連接到虛擬機器控制台。

4.7.2 集群命令

檢查集群狀態

說明：通過命令列檢查集群狀態

```
cluster status
```

檢查本地 CVM 服務狀態

說明：通過命令列檢查單個 CVM 的服務狀態

```
genesis status
```

Nutanix 集群升級

說明：通過命令列執行滾動(aka "live")集群升級

上傳升級包到一台 CVM 的~/tmp/
解壓縮升級包

```
tar xzvf ~/tmp/nutanix*
```



運行升級

```
tmp/install/bin/cluster -i ~/tmp/install upgrade
```

檢查狀態

```
upgrade status
```

節點升級

說明：運行升級將指定節點升級到當前集群版本

任意 CVM 上運行以下命令

```
cluster -u <NODE_IP(s)> upgrade_node
```

Hypervisor 升級狀態

說明：任意 CVM 運行命令檢查 Hypervisor 升級狀態

```
host_upgrade --status
```

詳細日誌(**on every CVM**)

```
~/data/logs/host_upgrade.out
```

命令列重啟集群及服務

說明：通過命令列重啟單一集群服務

停止服務

```
cluster stop <Service Name>
```

啟動服務

```
cluster start #NOTE: This will start all stopped services
```

通過命令列啟動集群服務

說明：命令列啟動已經停止的集群服務



啟動服務

```
cluster start #NOTE: This will start all stopped services
```

或者

啟動單一服務

```
Start single service: cluster start <Service Name>
```

命令列重啟本機服務

說明：命令列重啟單一集群服務

停止服務

```
genesis stop <Service Name>
```

啟動服務

```
cluster start
```

命令列啟動本機服務

說明：命令列啟動已經停止的集群服務

```
cluster start #NOTE: This will start all stopped services
```

通過命令列添加集群節點

說明：通過命令列添加集群節點

```
ncli cluster discover-nodes | egrep "Uuid" | awk '{print $4}' | xargs -I UUID
```

```
ncli cluster add-node node-uuid=UUID
```

顯示 vDisks 數量

說明：顯示 vDisks 數量

```
vdisk_config_printer | grep vdisk_id | wc -l
```



找到集群 ID

說明：找到當前集群 ID

```
zeus_config_printer | grep cluster_id
```

打開埠

說明：通過 IPTables 啟用埠

```
sudo vi /etc/sysconfig/iptables
```

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport <PORT> -j ACCEPT
```

```
sudo service iptables restart
```

檢查 Shadow Clones

說明：通過以下命令格式顯示 shadow clones : name#id@svm_id

```
vdisk_config_printer | grep '#'
```

重置延遲頁統計

說明：重置延遲頁(<CVM IP>:2009/latency)計數器

```
allssh "wget $i:2009/latency/reset"
```

查找 vDisks 數量

說明：查找 Distributed Storage Fabric (DFS) 上的 vDisk 數量

```
vdisk_config_printer | grep vdisk_id | wc -l
```

命令列開始 Curator 掃描

Description: Starts a Curator full scan from the CLI 說明：命令列開始 Curator 全掃描

```
allssh "wget -O - "http://$i:2010/master/api/client/StartCuratorTasks?task_type=2";"
```

壓縮環



說明：壓縮中繼資料環

```
allssh "nodetool -h localhost compact"
```

查找 **NOS** 版本

說明：查找 **NOS** 版本（注：也可使用 **NCLI**）

```
allssh "cat /etc/nutanix/release_version"
```

查找 **CVM** 版本

說明：查找 **CVM** 鏡像版本

```
llssh "cat /etc/nutanix/svm-version"
```

手工添加 **vDisk** 指紋

說明：產生一個指定的 **vdisk** 指紋（為重複資料刪除）注：重複資料刪除，必須在容器上啟用

```
vdisk_manipulator -vdisk_id=<vDisk ID> --operation=add_fingerprints
```

手動添加所有 **vDisk** 指紋

說明：為所有磁片生成指紋（用於重複資料刪除）注：重複資料刪除，必須在容器上啟用

```
for vdisk in `vdisk_config_printer | grep vdisk_id | awk '{print $2}'`; do vdisk_manipulator -vdisk_id=$vdisk --  
operation=add_fingerprints; done
```

顯示所有節點的 **Factory_Config.json** 文件

說明：顯示集群中所有節點的 **factory_config.json** 檔內容

```
allssh "cat /etc/nutanix/factory_config.json"
```

升級一個單一節點的 **NOS** 版本

說明：升級單一節點 **NOS** 版本匹配集群

```
cluster/bin/cluster -u <NEW_NODE_IP> upgrade_node
```



列出 DSF 文件 (vDisk)

說明：為了存儲 vDisks 列出 DSF 上的檔和相關資訊

```
Nfs_ls
```

獲取幫助

```
Nfs_ls --help
```

安裝 Nutanix Cluster Check(NCC)

說明：安裝 Nutanix Cluster Check (NCC) 健康檢查腳本來測試潛在的問題和集群健康

從 Nutanix Support Portal (portal.nutanix.com) 下載 NCC

SCP .tar.gz to the /home/nutanix directory

Untar NCC .tar.gz

```
tar xzmf <ncc .tar.gz file name> --recursive-unlink
```

運行安裝腳本

```
./ncc/bin/install.sh -f <ncc .tar.gz file name>
```

創建連結

```
source ~/ncc/ncc_completion.bash
```

```
echo "source ~/ncc/ncc_completion.bash" >> ~/.bashrc
```

運行 Nutanix Cluster Check (NCC)

說明：運行 Nutanix Cluster Check (NCC) 健康檢查腳本來測試潛在的問題和集群健康，集群故障時這是排查的第一步

確保 NCC 已經安裝（以上步驟）

運行 NCC health checks

```
ncc health_checks run_all
```

使用進程監控命令列列出任務



```
progress_monitor_cli -fetchall
```

使用進程監控命令列移除任務

```
progress_monitor_cli --entity_id=<ENTITY_ID> --operation=<OPERATION> --
entity_type=<ENTITY_TYPE> --delete
# NOTE: operation and entity_type should be all lowercase with k removed from the begining
```

4.7.3 指標和閾值

以下部分將涉及 Nutanix 後臺具體的指標和閾值。更多更新即將推出！

4.7.4 Gflags

4.7.5 故障排除和高級管理

查找 **Acropolis** 日誌

說明：查找 **Acropolis** 的集群日誌

```
allssh "cat ~/data/logs/Acropolis.log"
```

查找集群錯誤日誌

說明：為集群查找錯誤日誌

```
allssh "cat ~/data/logs/<COMPONENT NAME or *>.ERROR"
```

Stargate 示例

```
allssh "cat ~/data/logs/Stargate.ERROR"
```

查找集群致命日誌

說明：為集群查找致命日誌

```
allssh "cat ~/data/logs/<COMPONENT NAME or *>.FATAL"
```

Stargate 示例

```
allssh "cat ~/data/logs/Stargate.FATAL"
```

4.7.5.1 使用 2009 頁面(Stargate)

在大多數情況下，Prism 能夠提供所有需要的資訊和資料。然而，在某些情況下，如果需要一些更詳細的資料，你可以利用 Stargate 的 2009 頁。2009 頁可以通過導航到<CVM IP> : 2009 查看。

訪問後端頁

如果你在不同的網段 (L2 子網)，你需要添加一個訪問規則來訪問任何後端頁。

在頁面的頂部是概述細節顯示有關集群的各種詳細資訊：

Start time	2015-06-18 11:12:52-GMT-0700
Build version	el6-release-master-038d4c7d75cbc6ed21e64d357c94303350159807
Build last commit date	2015-05-30 14:14:03 -0700
Stargate handle	10.3.140.151:2009
iSCSI handle	10.3.140.151:3261
SVM id	7
Incarnation id	30986558
Highest allocated opid	38138394
Highest contiguous completed opid	36132415
Content cache total hits(NonDedup)	86.17%
Content cache flash pagin pct(NonDedup)	0
Content cache total hits(Dedup)	0%
Content cache flash pagin pct(Dedup)	0%
Content cache memory usage	3220 MB
Content cache physical memory usage	3224 MB
Content cache flash usage	0 MB
QoS Queue (size/admitted)	0/72
Oplog QoS queue (size/admitted)	0/0
NFS Flush Queue (size/admitted)	0/0
NFS cache usage	0 MB

圖. 2009 頁 - Stargate 概述

在本段有兩個區域要留意，第一個是 I/O 佇列，顯示 admitted / outstanding 操作的數量。

該圖顯示了概述的佇列部分：

QoS Queue (size/admitted)	0/26
Oplog QoS queue (size/admitted)	0/0

14-2 2009 頁 - Stargate 概述 – 仔列

第二部分是內容緩存的細節，顯示了快取記憶體大小的資訊和命中率該圖顯示了概述的內容緩存部分

Content cache flash pagin pct(NonDedup)	0
Content cache total hits(Dedup)	0%
Content cache flash pagin pct(Dedup)	0%
Content cache memory usage	3220 MB
Content cache physical memory usage	3224 MB
Content cache flash usage	0 MB

14-3 2009 頁 - Stargate 概覽 – 內容緩存

專家提示

在理想的情況下，如果工作負荷使用了最好的讀取性能命中率應高於 80%-90%+

注意：這些值是每個 Stargate / CVM

接下來的部分是“集群狀態”，顯示集群中不同的 Stargate 詳細資訊和它們的磁片使用率。

下圖顯示了 Stargate 和磁片利用率（可用/全部）：

SVM Id	IP:port	Incarnation	SSD-PCIe	SSD-SATA			DAS-SATA			
				154 (188/209)	153 (188/209)	152 (477/862)	151 (477/862)	150 (477/862)	149 (438/782)	
7	10.3.140.151:2009	30986558		146 (190/209)	145 (190/209)	144 (474/862)	143 (476/862)	142 (474/862)	141 (434/782)	
8	10.3.140.152:2009	30174288		50 (188/209)	49 (188/208)	48 (487/862)	47 (488/862)	44 (486/862)	43 (440/782)	
9	10.3.140.153:2009	30972235		139 (189/209)	138 (189/209)	137 (483/862)	136 (482/862)	135 (484/862)	134 (432/782)	
10	10.3.140.154:2009	30989925		90 (186/209)	89 (190/209)	88 (594/862)	87 (591/862)	86 (591/862)	85 (533/782)	
11	10.3.140.155:2009	30332545		123 (165/209)	122 (165/209)	121 (481/862)	120 (480/862)	119 (481/862)	117 (429/782)	
13	10.3.140.157:2009	30813522		78 (189/209)	77 (189/208)	76 (482/862)	75 (477/862)	74 (477/862)	73 (436/782)	
14	10.3.140.158:2009	30460780								

圖. 2009 頁 - 集群狀態 - 磁片使用情況

接下來的部分是“NFS Slave”部分，顯示每個虛擬磁片的各種詳細資訊和統計資訊。

下圖顯示了虛擬磁片和各種 I/O 的詳細資訊：

VDisk Name	Unstable data			Outstanding ops		Ops/s			KB/s		Avg latency (usec)		Avg op size	Avg outstanding	% busy
	KB	Ops/s	KB/s	Read	Write	Read	Write	Error	Read	Write	Read	Write			
NFS:31181822 (55b04a56-8e98-4f04-8c0f-617a57cb0450)	0	0	0	0	6	2248	907	0	8992	3628	178	2740	4096	5	85
NFS:31181826 (cde19589-df09-40a8-9640-6cad4e2d48bd)	0	0	0	1	5	2228	922	0	8912	3688	172	2756	4096	6	85
NFS:31182359 (0faec5e0-be91-40b8-9acf-5a3227f5c480)	0	0	0	0	0	0	0	0	0	0	0	0		0	0
NFS:31181823 (1a8ef41c-ac3f-4293-a46c-49d7e6c1c907)	0	0	0	0	6	2192	986	0	8768	3944	104	2198	4096	1	82
NFS:31181821 (813b97ae-99c1-4198-a3aa-791bd915b959)	0	0	0	0	5	2254	936	0	9016	3744	173	2761	4096	3	86
NFS:31181826 (bdc1e686-fb82-4157-9657-b8b038ab3e00)	0	0	0	0	0	0	0	0	0	0	0	0		0	0
NFS:31181824 (3d9bcd64-93de-4891-9351-500e589d2d2b)	0	0	0	0	3	2258	921	0	9032	3684	185	3243	4096	3	86
NFS:31181825 (4a3fc350-73a3-4cad-9104-4d5823143f48)	0	0	0	1	1	2289	956	0	9156	3824	133	2575	4096	3	84

圖. 14-5 2009 頁- NFS Slave - 虛擬磁片統計

專家提示

當看到任何潛在的性能問題請關注下幾點：

Avg. latency

Avg. op size

Avg. outstanding

欲瞭解更多具體細節請流覽 vdisk_stats 頁面。

4.7.5.2 使用 2009 / vdisk_stats 頁

2009 vdisk_stats 頁提供了每個虛擬磁片進一步的資料點。此頁面包括細節和隨機條帶圖，延遲條帶圖，I/O 大小和工作組的細節。

可以通過在左欄點擊“vDisk Id”導航到 vdisk_stats 頁面。

下圖顯示了部分和超連結的虛擬磁片 ID：

Hosted VDisks																					
VDisk Id	VDisk Name	Usage (GB)	Dedup (GB)	Oplog			Outstanding ops			Ops/s			KB/s		Avg latency (usec)	Avg op size	Avg qlen	% busy			
				KB	Fragments	Ops/s	KB/s	Read	Write	Estore	Read	Write	Error	Random	Read	Write					
15432793	NFS:20581239 (a849adb6-3809-423e-a89c-dd2fd2d665d6)	0	0	0	0	0	0	0,	0,	0KB	0	0	0	0	0	0	0	0	0		
31181823	NFS:31181823 (1a8ef41c-ac3f-4293-a46c-49d7e6c1c907)	2	0	198372	49593	990	3960	0,	0,	0KB	0	2192	1	0	2193	8768	320	46	4243	0	10
31181821	NFS:31181821 (813b97ae-99c1-4198-a3aa-791bd915b959)	2	0	196920	49230	939	3756	0,	0,	0KB	0	2253	0	0	2253	9012	0	38	4096	0	11

圖. 2009 頁 - 託管虛擬磁片

詳細的虛擬磁片統計。注：這些值是即時的，並且可以通過刷新頁面更新。



第一個關鍵區域是“Ops and Randomness”部分，顯示 I/O 模式是否在本質上隨機或順序。

該圖顯示了“操作和隨機性”部分：

VDisk 31181841 Ops and Randomness

	Read	Write	Total
IOPS (kIO/s)	2	1	3
IO Rate (MB/s)	9	4	13
Random %	100	100	
Sequential %	0	0	

圖. 2009 頁 - 虛擬磁片統計 – 操作和隨機性

下一個區域顯示前端讀寫的條帶圖和 I/O 延遲

該圖顯示了“前端讀取延遲”條帶圖：

VDisk 31181841 Frontend Read Latency

Latency Range	Average Bucket Latency	Number of Read IOs	Percent of Read IOs	Bar Graph
0 <= x < 1 ms	286	6211	92%	
1 ms <= x < 2 ms	1362	371	6%	
2 ms <= x < 5 ms	2855	135	2%	
5 ms <= x < 10 ms	5433	6	0%	
10 ms <= x < 20 ms				
20 ms <= x < 50 ms				
50 ms <= x < 100 ms				
100 ms <= x < inf				
Total	401	6723	100%	

14-8 2009 頁 - 虛擬磁片統計 - 前端讀取延遲

圖

該圖顯示了“前端寫入延遲”條帶圖：

VDisk 31181841 Frontend Write Latency

Latency Range	Average Bucket Latency	Number of Write IOs	Percent of Write IOs	Bar Graph
0 <= x < 1 ms				
1 ms <= x < 2 ms	1724	167	6%	
2 ms <= x < 5 ms	3422	2098	72%	
5 ms <= x < 10 ms	6369	562	19%	
10 ms <= x < 20 ms	12297	88	3%	
20 ms <= x < 50 ms	23386	6	0%	
50 ms <= x < 100 ms				
100 ms <= x < inf				
Total	4200	2921	100%	

圖

14-9

2009 頁 - 虛擬磁片統計 - 前端寫延遲

接下來的關鍵區域是 I/O 大小分佈，顯示讀取的條帶圖和寫入 I/O 的大小。

該圖顯示了“Read Size Distribution”條帶圖：

VDisk 31181841 Read Size Distribution

Latency Range	Average Bucket Size	Number of Read IOs	Percent of Read IOs	Bar Graph
0 <= x < 4 kB				
4 kB <= x < 8 kB	4096	6723	100%	
8 kB <= x < 16 kB				
16 kB <= x < 32 kB				
32 kB <= x < 64 kB				
64 kB <= x < 512 kB				
512 kB <= x < 1 MB				
1 MB <= x < inf				
Total	4096	6723	100%	

圖

14-10

2009 頁 - 虛擬磁片統計 - 讀 I/O 大小

下圖顯示了“Write Size Distribution”條帶圖：

VDisk 31181841 Write Size Distribution

Latency Range	Average Bucket Size	Number of Write IOs	Percent of Write IOs	Bar Graph
0 <= x < 4 kB				
4 kB <= x < 8 kB	4096	2921	100%	
8 kB <= x < 16 kB				
16 kB <= x < 32 kB				
32 kB <= x < 64 kB				
64 kB <= x < 512 kB				
512 kB <= x < 1 MB				
1 MB <= x < inf				
Total	4096	2921	100%	

圖

14-11

2009 頁 - 虛擬磁片統計 - 寫 I/O 大小

接下來的重點區域是“Working Set Size”部分，提供有關 Working Set Size 的最後 2 分鐘和 1 小時的詳細資訊。細分為讀寫 I/O。

下圖顯示了“Working Set Sizes”表：

VDisk 31181841 Working Set Sizes

	2 min	1 hr
Read WSS (MB)	2030	0
Write WSS (MB)	2030	0
Union WSS (MB)	2030	0

圖. 2009 頁 - 虛擬磁片統計 - Working Set Sizes



“Read Source”提供該層或位置的讀 I/O 資訊。

下圖顯示了“Read Source”的詳細資訊：

VDisk 31181841 Read Source

Source	Data (MB/s)	Percent
Oplog		
Extent cache		
Cache DRAM	7	75%
Cache SSD	2	25%
Estore SSD		
Estore HDD		
Block Store		
Zeroes		
Total	9	100%

圖. 2009 頁 - 虛擬磁片統計 - Read Source

專家提示

如果您看到高的讀取延遲，看一下虛擬磁片的讀源，並看一下 I/O 的服務源。在大多數情況下，高延遲可以通過讀取來自 HDD (Estore HDD) 引起的。

“Write Destination”部分將顯示其中新寫入的 I/O。

該圖顯示了“Write Destination”表：



VDisk 31181841 Write Destination

Destination	Data (MB/s)	Percent
Oplog	4	100%
Estore		
Block Store		
Total	4	100%

圖. 2009 頁 - 虛擬磁片統計 - Write Destination

專家提示

隨機或更小的 I/O (<64K) 將被寫入到 OPLOG。較大的或順序的 I/O 將繞過 OPLOG 和直接寫到盤區存儲 (ESTORE)。

另一個有趣的資料是什麼資料通過 ILM 被從 HDD 硬碟遷移到 SSD 固態硬碟。在 'Extent Group Up-Migration "表顯示，在過去 300、3600 和 86400 秒已經遷移的資料。

下圖顯示了“Extent Group Up-Migration ”表：

VDisk 31181841 Extent Group Up-Migration

Last 300 seconds	0
Last 3600 seconds	18
Last 86400 seconds	18

圖. 2009 頁 - 虛擬磁片統計 - Extent Group Up-Migration

4.7.5.3 使用 2010 頁 (Curator)



2010 頁是用於監控 Curator MapReduce 框架的詳細頁面。該頁面提供了作業、掃描和相關任務的詳細資訊。

您可以通過導航到 <http://<CVM IP>:2010>。注意：如果你不是 Curator Master 請點擊“Curator Master:”後的 IP。

頁面的頂部將顯示 Curator Master 細節包括正常執行時間、版本等等。

下一節是“Curator Nodes”表，它顯示有關集群中的節點、角色和健康狀況的各種細節。這些節點將被用於分散式進程和選舉任務。

該圖顯示了“Curator Nodes”表：

Curator Nodes

Sl. No.	IP:port	Type	Node	Incarnation Id	MapReduce Version	Build Version	Curator Disks	Health Status
1	10.3.140.151:2010	Master	357	30058470	42000160	159807	1	Healthy
2	10.3.140.152:2010	Slave	319	30174391	42000160	159807	1	Healthy
3	10.3.140.153:2010	Slave	360	30972348	42000160	159807	1	Healthy
4	10.3.140.154:2010	Slave	356	30661501	42000160	159807	1	Healthy
5	10.3.140.155:2010	Slave	318	30332648	42000160	159807	1	Healthy
6	10.3.140.157:2010	Slave	321	30813635	42000160	159807	1	Healthy
7	10.3.140.158:2010	Slave	322	30491731	42000160	159807	1	Healthy

圖. 2010 頁 - Curator Nodes

接下來的部分是“Curator jobs”表，該表顯示已完成或當前正在運行的作業。作業有兩種主要類型，部分掃描是每隔 60 分鐘，全面掃描是每 6 小時。注意：基於利用率和其他活動的時間是可變的。

這些掃描將定期調度運行，也可以由某些集群事件觸發。

這裡有一些工作執行的原因：

- 定期（正常狀態下）
- 磁片/節點/主機殼失效
- ILM 失衡
- 磁片/層失衡

下圖顯示“Curator Jobs”表：



Curator Jobs

Job id	Execution id	Job name	Status	Reasons	Background tasks			Start time	End time	Total time (secs)	Scan timeout (secs)
					Submitted	Canceled	Generated				
1	21063	Partial Scan	Succeeded	Periodic	0	0	0	Jul 13 13:59:14	Jul 13 14:05:12	358	43200
1	21061	Partial Scan	Succeeded	Periodic	0	0	0	Jul 13 12:53:07	Jul 13 12:59:13	366	43200
1	21059	Partial Scan	Succeeded	Periodic	0	0	0	Jul 13 11:46:33	Jul 13 11:53:06	393	43200
0	21054	Full Scan	Succeeded	Periodic	34	0	34	Jul 13 10:29:30	Jul 13 10:46:32	1022	43200
1	21052	Partial Scan	Succeeded	Periodic	0	0	0	Jul 13 09:55:01	Jul 13 10:01:12	371	43200
1	21050	Partial Scan	Succeeded	Periodic	0	0	0	Jul 13 08:48:44	Jul 13 08:55:00	376	43200
1	21048	Partial Scan	Succeeded	Periodic	0	0	0	Jul 13 07:42:04	Jul 13 07:48:43	399	43200
1	21046	Partial Scan	Succeeded	Periodic	0	0	0	Jul 13 06:36:08	Jul 13 06:42:03	355	43200
1	21044	Partial Scan	Succeeded	Periodic	6	0	6	Jul 13 05:29:30	Jul 13 05:36:07	397	43200
0	21039	Full Scan	Succeeded	Periodic	37	0	37	Jul 13 04:12:12	Jul 13 04:29:29	1037	43200

圖. 2010 頁 - Curator Jobs
該表顯示了一些由每個作業進行的高級別活動

Table 0. Curator 掃描任務

活動	全掃描	局部掃描
ILM	X	X
磁片平衡	X	X
壓縮	X	X
重複資料刪除	X	
擦除編碼	X	
垃圾清理	X	

點擊“Execution id”將帶您到作業詳細資訊頁面，該頁面顯示各種統計工作，以及所產生的任務。

在頁面的頂部表將顯示在工作中的各種資訊的類型、原因、任務和持續時間。

接下來的部分是“幕後工作統計”表，該表上的任務類型顯示各種資訊，產生量和優先順序。

下圖中顯示作業詳細資訊表：

Job id	0
Execution id	21054
Scan Type	Full
Reasons	Periodic
Bg tasks generated	34
Bg tasks submitted	34
Bg tasks cancelled	0
Start time	Jul 13 10:29:30
End time	Jul 13 10:46:32
Total time (secs)	1022
Scan timeout (secs)	43200

圖. 2010 頁 - Curator Job – 詳細資訊

該圖顯示了“幕後工作統計”表：

Background Task Stats

Job Name	Generated				Submitted				Cancelled			
	High	Medium	Low	Total	High	Medium	Low	Total	High	Medium	Low	Total
MigrateExtents	0	0	1	1	0	0	1	1	0	0	0	0
UpdateRefcounts	0	33	0	33	0	33	0	33	0	0	0	0
Totals	0	33	1	34	0	33	1	34	0	0	0	0

圖. 2010 頁- Curator Job – 任務

接下來的部分是“MapReduce 任務”表，該表顯示了每個 Curator job 開始實際的 MapReduce 工作。局部掃描將有一個單一的 MapReduce 工作，全面掃描，將有四個 MapReduce 工作。

下圖顯示“MapReduce Jobs”表

MapReduce Jobs

Job id	Job name	Status	Map tasks done	Reduce tasks done	Fg tasks	Bg tasks	Errors	Start time	End time	Total time (secs)
21064	PartialScan MapReduce	Succeeded	35/35	35/35	2493	0	0	Jul 13 14:00:14	Jul 13 14:05:12	298
21062	PartialScan MapReduce	Succeeded	35/35	35/35	2492	0	0	Jul 13 12:54:07	Jul 13 12:59:12	305
21060	PartialScan MapReduce	Succeeded	35/35	35/35	2493	0	0	Jul 13 11:47:34	Jul 13 11:53:05	331
21058	FullScan MapReduce #4	Succeeded	14/14	28/28	2621	34	0	Jul 13 10:41:13	Jul 13 10:46:30	317
21057	FullScan MapReduce #3	Succeeded	7/7	7/7	0	0	0	Jul 13 10:38:26	Jul 13 10:41:13	167
21056	FullScan MapReduce #2	Succeeded	7/7	7/7	0	0	0	Jul 13 10:33:47	Jul 13 10:38:26	279
21055	FullScan MapReduce #1	Succeeded	15/15	14/14	33	0	0	Jul 13 10:33:30	Jul 13 10:33:47	17
21053	PartialScan MapReduce	Succeeded	35/35	35/35	2493	0	0	Jul 13 09:56:01	Jul 13 10:01:11	310
21051	PartialScan MapReduce	Succeeded	35/35	35/35	2492	0	0	Jul 13 08:49:45	Jul 13 08:54:59	314
21049	PartialScan MapReduce	Succeeded	35/35	35/35	2492	0	0	Jul 13 07:43:04	Jul 13 07:48:42	338

圖. 2010 頁- MapReduce 作業

點擊“Job ID”將帶您到 MapReduce 工作的詳細資訊頁面，該頁面會顯示任務的狀態，各種計數器以及關於 MapReduce 工作細節。

圖中顯示了一些工作的計數器的示例：

Job Counters

Name	Value
MapExtentGroupIdMap	2155990
MapExtentGroupAccessDataMap	2155985
NumExtentGroupsToMigrateForILM	0
NumExtentGroupsToMigrateForDiskBalancing	0
NumTasksToMigrateErasureCodedExtents	0
ReduceNonDedupExtentIdTrueRefCount	13023596
ReduceDedupExtentIdTrueRefCount	3295838
ReduceNonDedupExtentIdRefCount	13037299
ReduceDedupExtentIdRefCount	3295838
ReduceDedupExtentGroupId	2752217

圖. 2010 頁- MapReduce 作業-計數器



主頁上的下一個部分是“Queued Curator Jobs”和“Last Successful Curator Scans”部分。這些表顯示當定期掃描正常運行時，最後一次成功掃描的詳細資訊。

該圖顯示了“Curator 作業佇列”和“Curator 最後一次成功掃描”部分：

Queued Curator Jobs

Ring change in progress? No

Job id	Job name	Eligible time (secs)
1	Partial Scan	2516
0	Full Scan	8596

Last Successful Curator Scans

Job name	Start time	End time	Total time (secs)	Master handle	Incarnation id	Job id	Execution id	Status	Reasons	Num MR jobs
Partial Scan	Jul 13 13:59:14	Jul 13 14:05:12	358	10.3.140.151:2010	30058470	1	21063	Succeeded	Periodic	1
Full Scan	Jul 13 10:29:30	Jul 13 10:46:31	1021	10.3.140.151:2010	30058470	0	21054	Succeeded	Periodic	4

圖. 2010 頁 - 佇列和成功掃描

4.7.5.4 高級命令列說明

Prism should provide all that is necessary in terms of normal troubleshooting and performance monitoring. However, there may be cases where you want to get more detailed information which is exposed on some of the backend pages mentioned above, or the CLI.

Prism 提供所有用於普通排錯和性能監控的必須資訊。但是可能有些事件中，你需要拿到更多在某些後端頁面顯示的詳細資訊或者命令列。

vdisk_config_printer

`vdisk_config_printer` 命令顯示集群中所有 `vdisk` 的資訊清單。

比較重要的包括：

- Vdisk ID
- Vdisk name
- Parent vdisk ID (if clone or snapshot)
- Vdisk size (Bytes)
- Container id
- To remove bool (to be cleaned up by curator scan)
- Mutability state (mutable if active r/w vdisk, immutable if snapshot)

以下是示例輸出：

```
nutanix@NTNX-13SM35210012-A-CVM:~$ vdisk_config_printer | more
...
vdisk_id: 1014400
vdisk_name: "NFS:1314152"
parent_vdisk_id: 16445
vdisk_size: 400000000000
container_id: 988
to_remove: true
creation_time_usecs: 1414104961926709
mutability_state: kImmutableSnapshot
closest_named_ancestor: "NFS:852488"
vdisk_creator_loc: 7
vdisk_creator_loc: 67426
vdisk_creator_loc: 4420541
nfs_file_name: "d12f5058-f4ef-4471-a196-c1ce8b722877"
may_be_parent: true
parent_nfs_file_name_hint: "d12f5058-f4ef-4471-a196-c1ce8b722877"
last_modification_time_usecs: 1414241875647629
...
```

vdisk_usage_printer -vdisk_id=<VDISK ID>

vdisk_usage_printer 用來獲取 vdisk, 相關 extents 和 egroups 的詳細資訊。

比較重要的包括：

- Egroup ID
- Egroup extent count
- Untransformed egroup size
- Transformed egroup size
- Transform ratio
- Transformation type(s)
- Egroup replica locations (disk/cvm/rack)

以下是示例輸出：

```
nutanix@NTNX-13SM35210012-A-CVM:~$ vdisk_usage_printer -vdisk_id=99999
    Egid # eids UT Size T Size ... T Type Replicas(disk/svm/rack)
1256878 64 1.03 MB 1.03 MB ... D,[73 /14/60][184108644/184108632/60]
1256881 64 1.03 MB 1.03 MB ... D,[73 /14/60][152/7/25]
1256883 64 1.00 MB 1.00 MB ... D,[73 /14/60][184108642/184108632/60]
1055651 4 4.00 MB 4.00 MB ... none[76 /14/60][184108643/184108632/60]
1056604 4 4.00 MB 4.00 MB ... none[74 /14/60][184108642/184108632/60]
1056605 4 4.00 MB 4.00 MB ... none[73 /14/60][152/7/25]
...

```

注：用於去重的 **egroup** 大小和非去重的 **egroups** 大小分別為（1 vs. 4MB）。 “資料結構”章節提到過，對於去重的資料，1MB 的 **egroup** 有利於消除去重資料導致的可能的碎片。

curator_cli display_data_reduction_report

curator_cli display_data_reduction_report 用來獲取轉換產生的容器存儲節省的空間的詳細資訊（例如 clone, snap, dedup, compression, erasure coding 等等）。

比較重要的包括：

- Container ID
- Technique (transform applied)
- Pre reduction Size
- Post reduction size
- Saved space
- Savings ratio

以下是示例輸出：

CVM:~\$ curator_cli display_data_reduction_report						
Using curator master: 99.99.99.99:2010						
Using execution id 68188 of the last successful full scan						
+-----+-----+-----+-----+-----+-----+-----+						
Container	Technique	Pre Reduction	Post Reduction	Saved	Ratio	
ID						
+-----+-----+-----+-----+-----+-----+-----+						
988	Clone	4.88 TB	2.86 TB	2.02 TB	1.70753	
988	Snapshot	2.86 TB	2.22 TB	656.92 GB	1.28931	
988	Dedup	2.22 TB	1.21 TB	1.00 TB	1.82518	
988	Compression	1.23 TB	1.23 TB	0.00 KB	1	
988	Erasure Coding	1.23 TB	1.23 TB	0.00 KB	1	
26768753	Clone	764.26 GB	626.25 GB	138.01 GB	1.22038	
26768753	Snapshot	380.87 GB	380.87 GB	0.00 KB	1	
84040	Snappy	810.35 GB	102.38 GB	707.97 GB	7.91496	
6853230	Snappy	3.15 TB	1.09 TB	2.06 TB	2.88713	
12199346	Snappy	872.42 GB	109.89 GB	762.53 GB	7.93892	
12736558	Snappy	9.00 TB	1.13 TB	7.87 TB	7.94087	
15430780	Snappy	1.23 TB	89.37 GB	1.14 TB	14.1043	
26768751	Snappy	339.00 MB	45.02 MB	293.98 MB	7.53072	
27352219	Snappy	1013.8 MB	90.32 MB	923.55 MB	11.2253	
+-----+-----+-----+-----+-----+-----+-----+						

curator_cli get_vdisk_usage lookup_vdisk_ids=<COMMA SEPARATED VDISK ID(s)>

curator_cli display_data_reduction_用來獲取每種轉換產生的容器存儲節省的空間的詳細資訊（例如 clone, snap, dedup, compression, erasure coding 等等）。

比較重要的包括：

- Vdisk ID
- Exclusive usage (Data referred to by only this vdisk)
- Logical uninherited (Data written to vdisk, may be inherited by a child in the event of clone)
 - Logical dedup (Amount of vdisk data that has been deduplicated)
 - Logical snapshot (Data not shared across vdisk chains)
 - Logical clone (Data shared across vdisk chains)

以下是示例輸出：

```
Using curator master: 99.99.99.99:2010
```

VDisk usage stats:

VDisk Id	Exclusive usage	Logical Uninherited	Logical Dedup	Logical Snapshot	Logical Clone	Logical
254244142	596.06 MB	529.75 MB	6.70 GB	11.55 MB	214 MB	
15995052	599.05 MB	90.70 MB	7.14 GB	0.00 KB	4.81 MB	
203739387	31.97 GB	31.86 GB	24.3 MB	0.00 KB	4.72 GB	
22841153	147.51 GB	147.18 GB	0.00 KB	0.00 KB	0.00 KB	
...						

curator_cli get_egroup_access_info

`curator_cli get_egroup_access_info` 用來獲取基於最後訪問（讀）/修改（覆蓋寫或者寫）的每個桶的 `egroups` 的數量。該資訊用來評估適合用於糾刪碼的候選的 `egroup` 的數量。

比較重要的包括：

- Container ID
- Access \ Modify (secs)

以下是示例輸出：

```
Using curator master: 99.99.99.99:2010
```

Container Id: 988

Access \ Modify (secs)	[0,300]	[300,3600]	[3600,86400]	[86400,604800]	[604800,2592000]	[2592000,15552000]	[15552000,inf)
[0,300)	345	1	0	0
[300,3600)	164	817	0	0
[3600,86400)	4	7	3479	7
[86400,604800)	0	0	81	7063
[604800,2592000)	0	0	15	22
[2592000,15552000)	1	0	0	10
[15552000,inf)	0	0	0	1
...							
...							

5 第五部分：AHV

5.1 架構

5.1.1 節點架構

在 AHV 的部署中，控制器虛擬機器（CVM）以虛擬機器形式運行，並通過 PCI 直通方式訪問磁片。這樣就可以讓 CVM 繞過虛擬化層直接控制 PCI 設備。AHV 是基於 CentOS KVM 基礎開發的，採用全硬體虛擬化路線實現。

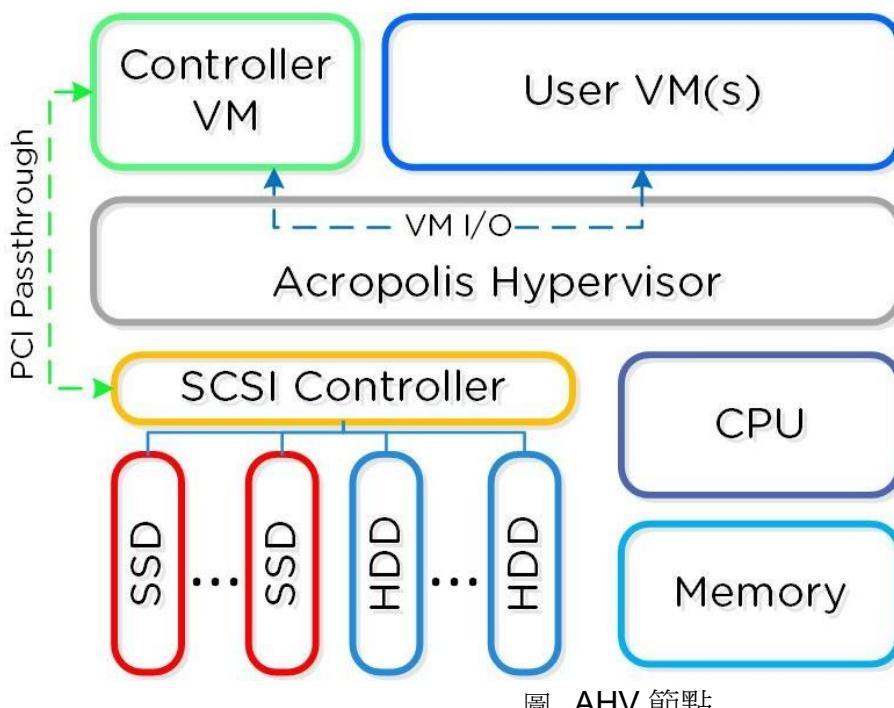


圖. AHV 節點

AHV 虛擬化以 CentOS KVM 為基礎並擴展了諸多高級功能，如 HA，線上遷移等。同時，AHV 虛擬化系統也通過了微軟伺服器虛擬化驗證程式的認證，可以運行微軟的作業系統及應用程式。

5.1.2 KVM 架構

KVM 包括以下的主要組件：

- KVM-kmod
 - KVM 內核模組
- Libvirt
 - 一個 API，管理 KVM 和 QEMU 的後臺進程和管理工具。負責 Acropolis 與 KVM / QEMU 通過 libvirtd 進程進行通訊
 - Qemu-kvm

◦ 一個模擬器（machina emulator），使得每一個虛擬機器能夠獨立運行。AHV 通過它來實現硬體輔助虛擬化，並同樣支持虛擬機器以全虛擬化HVM 的形式運行。下圖顯示了各個元件之間的關係：

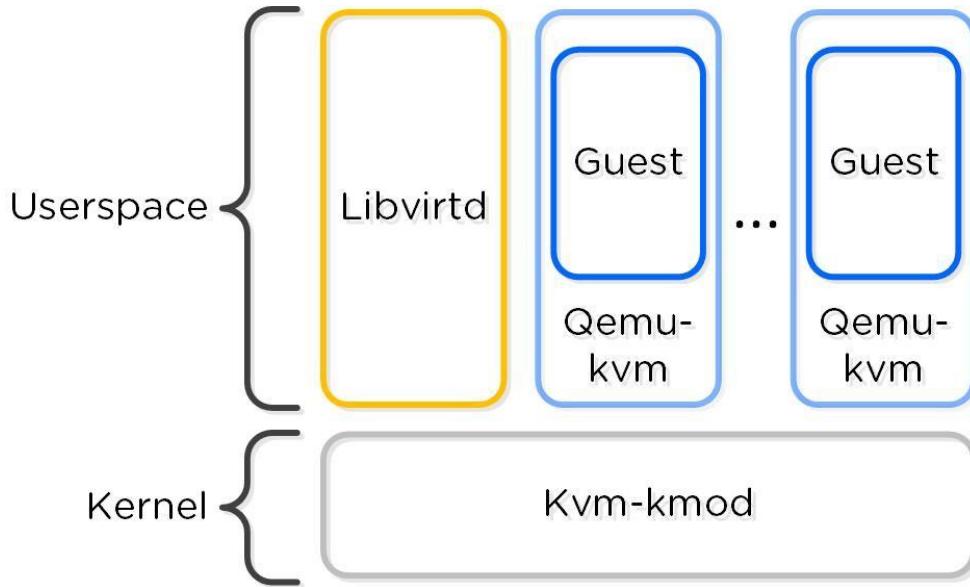


圖 . KVM 組件關係

Acropolis 與 KVM 之間通過 Libvirtd 進行通迅。

處理器相容性

類似于 VMware 的增強 vMotion 相容性(EVC) 功能，允許虛擬機器在不同代處理器之間進行移動；AHV 將自動檢測集群中最低的處理器，並限制所有的 QEMU 域運行在這個級別中。通過此功能允許包含跨代處理器的 AHV 集群中，提供虛擬機器在主機間線上遷移的能力。

5.1.3 最高配置和可擴展性

以下是配置的最大限制及範圍

- 集群最大數量 (Maximum cluster size) : N/A – 跟 Nutanix 集群一樣
- 每個 VM 的最大虛擬 CPU (Maximum vCPUs per VM) : 每台主機的最大物理核數
 - 每個 VM 的最大記憶體 (Maximum memory per VM) : 4TB 或者可用實體記憶體中的最小值
 - 每個虛擬磁片的容量: **9EB* (Exabyte)**
 - 每個節點的最大 VM 數量 (Maximum VMs per host) : N/A – 取決於記憶體的分配限制
 - 每個集群的最大 VM 數量 (Maximum VMs per cluster) : N/A –取決於記憶體的分配限制

*AHV 不像 ESXi / Hyper-V 這樣傳統的存儲堆疊; 在 AHV 中，所有的磁片以裸 SCSI 塊設備方式直通到虛擬機器中。這意味著最大的虛擬磁片容量僅受限於 DSF 的虛擬磁片容量(最大 9 EB)。

5.1.4 網路

AHV 為所有虛機網路採用了 Open vSwitch (OVS)。虛機聯網是通過 Prism / ACLI 配置的，每個虛擬機器網卡都連接到一個 Tap 介面。

下圖展示了 OVS 架構的概念圖：

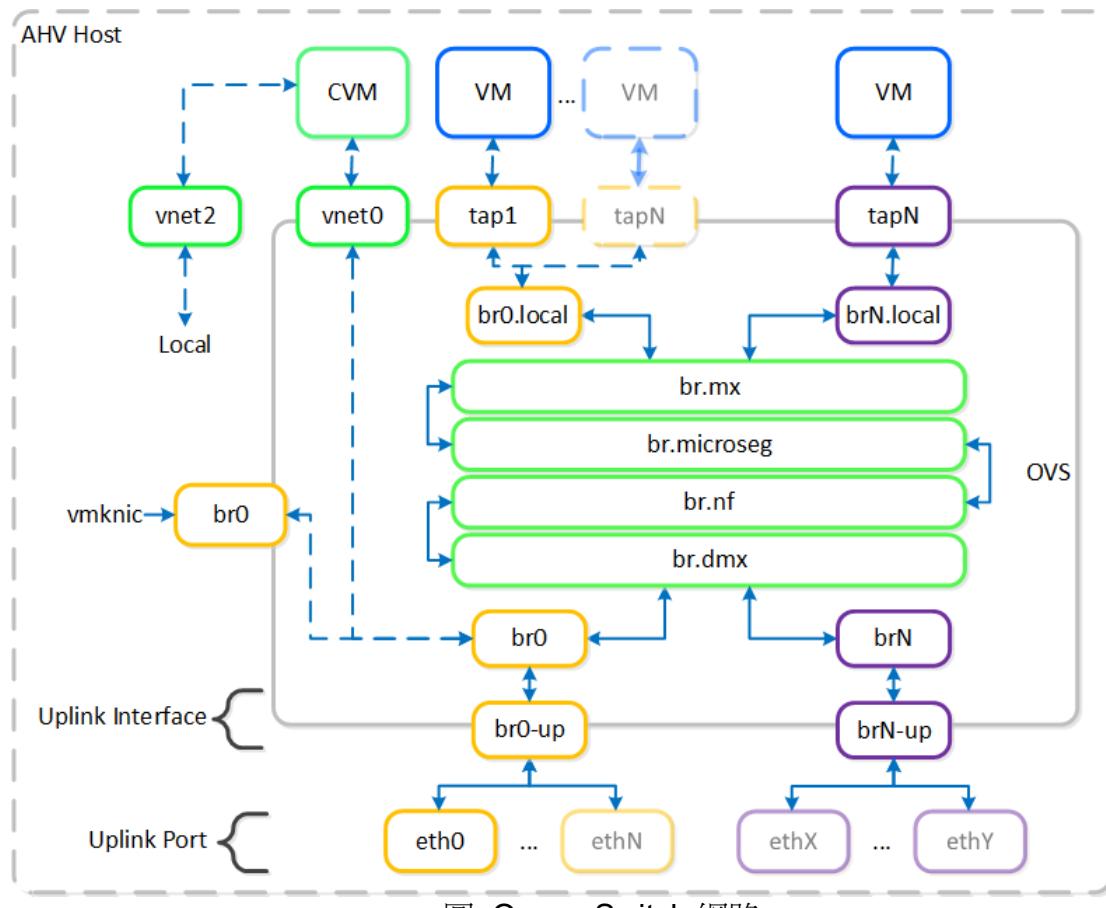


圖. Open vSwitch 網路

在前面的架構圖中，你可以看到一些類型的元件：

Open vSwitch (OVS)



OVS 是一個開源軟體交換機，在 Linux 內核中實現，設計用於多伺服器虛擬化環境。預設情況下，OVS 的行為類似於維護 MAC 位址表的第 2 層學習交換機。

Hypervisor 主機和虛機接到交換機上的虛擬埠。

OVS 支援許多流行的交換機特性，包括 VLAN 標記、鏈路聚合控制協議 (LACP)、埠鏡像和服務品質(QoS)等。每個 AHV 伺服器維護一個 OVS 實例，所有 OVS 實例組合在一起形成一個邏輯開關。稱為橋接器的結構會管理駐留在 AHV 主機上的交換機實例。

Bridge

橋接器充當虛擬交換機，管理物理和虛擬網路介面之間的網路通信通路。預設的 AHV 配置包括一個名為 br0 的 OVS 橋接器和一個名為 virbr0 的本地 Linux 橋接器。virbr0 Linux 橋接器在 CVM 和 AHV 主機之間進行管理和存儲通信。所有其他存儲、主機和 VM 網路流量都通過 br0 OVS 橋接器。AHV 主機、虛機和物理介面使用“Port”來連接到橋接器。

Port

埠是在表示到虛擬交換機的連線性的橋上創建的邏輯結構。Nutanix 使用幾種埠類型，包括內部埠、Tap、VXLAN 和 Bond:

- 內部埠 - 與默認橋接器(br0)同名的內部埠為 AHV 主機提供訪問。
- Tap 埠 - 充當虛擬網卡到虛機的橋樑連接。
- VXLAN 埠 - 用於 Acropolis 提供的 IP 地址管理功能。
- 繩定埠 - 為 AHV 主機的物理介面提供網卡組。

Bond

繩定埠聚合 AHV 主機上的物理介面。預設情況下，名為 br0-up 的繩定組是在 bridge br0 中創建的。在節點創建映射過程後，所有的介面都放在一個繩定組內，這是 Foundation 創建映射的過程的要求。對默認繩定埠 br0-up 的更改通常將其重命名為 bond0。Nutanix 建議使用名稱 br0-up 來快速識別作為橋接 br0 上行鏈路的介面。



OVS 綁定允許多種負載平衡模式，包括活動 active-backup、balance-slb 和 balance-tcp。LACP 也可以在綁定埠啟動。在安裝期間沒有指定“bond_mode”設置，默認為 active-backup，這是推薦的配置。

5.1.4.1 上行鏈路負載平衡

在前一節中簡要提到，跨綁定組上行鏈路的流量平衡是可能的。

以下鍵合模式可供選擇：

- **active-backup**
 - 在一個活動適配器上傳輸所有流量的預設配置。如果活動適配器變得不可用，則綁定中的另一個適配器將變為活動的。限制輸送量最大可達單個網口的帶寬。(推薦)
- **balance-slb**

在連接的適配器之間平衡每個虛機的網卡 (例如 VM A nic 1 - eth0 / nic 2 - eth1)。將虛擬機器網口的輸送量限制為單個網口的頻寬，但是具有 x 個網口的虛擬機器可以利用 x * 適配器頻寬(假設 x 對於綁定中的虛機網口和物理上行適配器的數量是相同的)。注意：對於多播流量有一些注意要求
- **balance-tcp / LACP**

在綁定的適配器之間平衡每個虛機網口的 TCP 會話。將每個網口的輸送量限制為最大連接頻寬(物理上行適配器數量 * 速度)。需要連結聚合，並在需要 LACP 時使用。

你可以在 AHV Networking 手冊中找到綁定模式的其他資訊 ([LINK](#)).

5.1.4.2 VM 網卡類型

AHV 支援以下兩種虛擬機器網路介面類別型：

- Access(默認)
- Trunk(4.6 版本以及上)

預設情況下，虛擬機器的網卡以 Access 模式創建(類似於您在埠組中所看到的虛擬機器網卡)，但是它也可以以 Trunk 介面方式連接到虛擬機器作業系統。

Trunk 介面能通過以下命令列添加：

```
vm.nic_create <VM_NAME> vlan_mode=kTrunked trunked_networks=<ALLOWED_VLANS>
network=<NATIVE_VLAN>
```

示例：

```
vm.nic_create fooVM vlan_mode=kTrunked trunked_networks=10,20,30 network=vlan.10
```

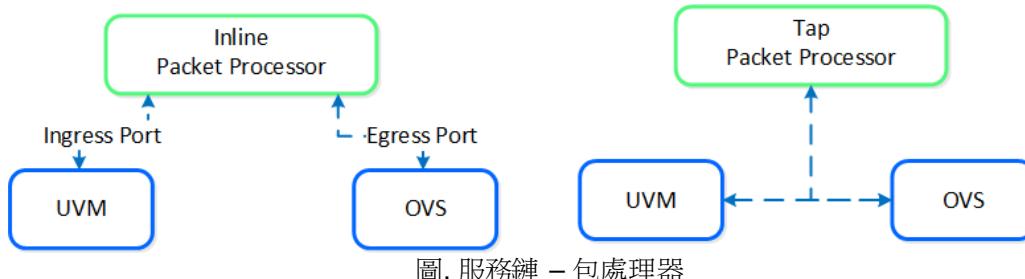
5.1.4.3 Service Chaining 服務鏈

作為網路路徑的一部分，AHV 服務鏈功能允許截取所有流量，並轉發給資料包處理器（NFV，設備，虛擬裝置等）用於後續處理。

服務鏈的常見用途：

- 防火牆（如 Palo Alto 等）
- 負載均衡器（例如 F5，NetScaler 等）
- IDS / IPS / 網路監視器

在服務鏈中有兩種類型的包處理器：



- 即時包處理器
 - 資料包通過 OVS 時即時攔截
 - 可以修改和允許/拒絕資料包
 - 常見用途：防火牆和負載均衡器
- 觸發式包處理器
 - 在資料包流動時檢查，只能讀取資料包流中的資料
 - 常見用途：IDS / IPS / 網路監視器

任何服務連結都是在應用流-微分割規則之後，並在包離開本地 OVS 之前完成的。這發生在網路功能橋上(br.nf)：

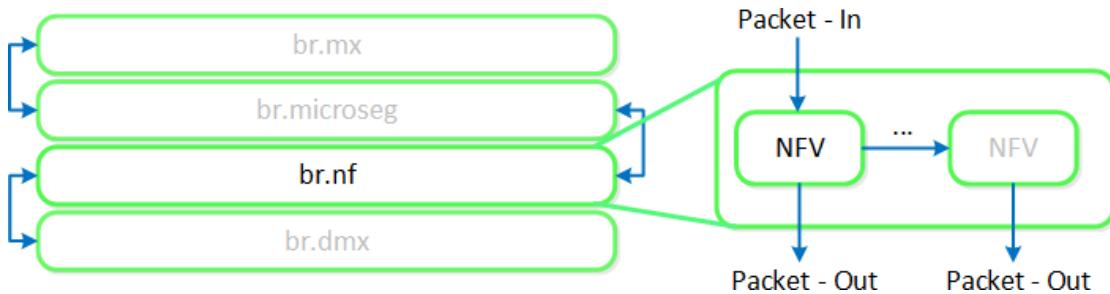


圖.服務鏈 – Flow

注意:可以在一條鏈中將多個 NFV /包處理器串在一起。

5.2 如何工作

5.2.1 存儲 I/O 路徑

AHV 不像 ESXi / Hyper-V 這樣傳統的存儲堆疊; 在 AHV 中，所有的磁片以裸 SCSI 塊設備方式直通到虛擬機器中。這樣能夠保持最佳和最輕量的 I/O 路徑。

專家提示

Acropolis 為最終用戶抽象 kvm, virsh, qemu, libvirt 和 iSCSI，並處理所有後端配置。這允許用戶通過 Prism / ACCLI 將焦點集中在虛擬機器上。以下僅供參考，不建議使用 virsh, libvirt 等進行手動處理。

每一個 KVM 主機都有一個 iSCSI 重定向守護程式通過 NOP OUT 命令來檢查 Stargate 的健康情況。

- o 在 iscsi_redirector 日誌中 (在 AHV 主機/var/log/的目錄中), 可以看到每個 Stargate 的健康狀況:

```
2017-08-18 19:25:21,733 - INFO - Portal 192.168.5.254:3261 is up
...
2017-08-18 19:25:25,735 - INFO - Portal 10.3.140.158:3261 is up
2017-08-18 19:25:26,737 - INFO - Portal 10.3.140.153:3261 is up
```

注: 本地 Stargate 通過它的內部位址 192.168.5.254 顯示

通過以下輸出可以看出 `iscsi_redirector` 監聽 127.0.0.1:3261 埠

```
[root@NTNX-BEAST-1 ~]# netstat -tnlp | egrep tcp.*3261  
Proto ... Local Address Foreign Address State PID/Program name  
...  
tcp ... 127.0.0.1:3261 0.0.0.0:* LISTEN 8044/python  
...
```

QEMU 配置 iSCSI Redirector 作為 iSCSI 目標門戶。在有登錄請求時，iSCSI Redirector 將把 iSCSI 登錄請求重定向到一個健康的 Stargate（默認本地優先）。

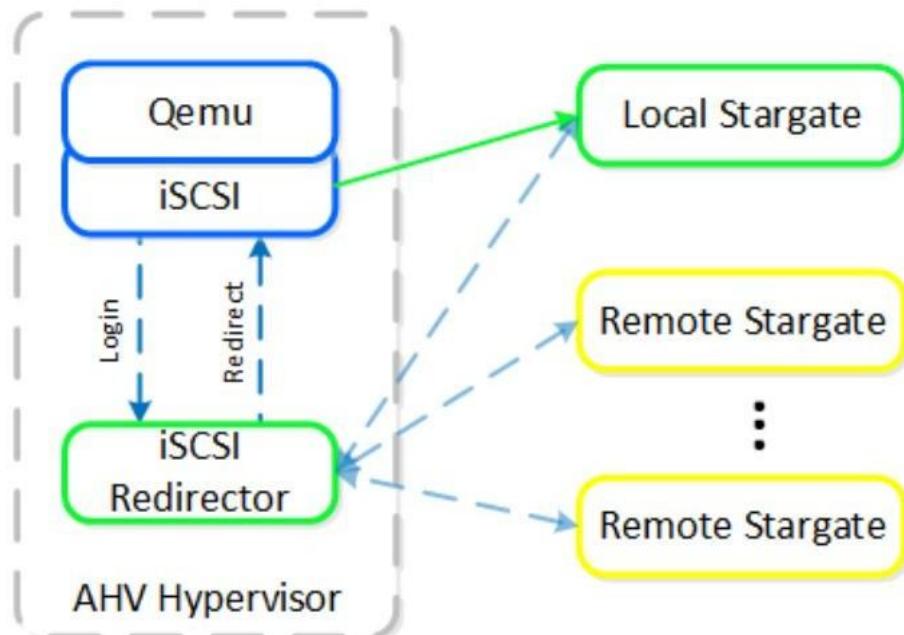


圖 13-4 iSCSI 多路徑-正常狀態

請通過域 XML 檔查看配置：

```
<devices>
```



```
<emulator>/usr/libexec/qemu-kvm</emulator>

...
<disk type='network' device='lun'>

    <driver name='qemu' type='raw' cache='none' error_policy='report' io='native'/>

    <source protocol='iscsi' name="iqn.2010-06.com.nutanix:vmdisk-16a5..0'>

        <host name='127.0.0.1' port='3261'/>

    </source>

    <backingStore/>

    <target dev='sda' bus='scsi'>

        <boot order='1'>

            <alias name='scsi0-0-0-0'>

                <address type='drive' controller='0' bus='0' target='0' unit='0'>

            </address>
        </target>
    ...
</disk>

</devices>
```

首選的控制器類型是 **virtio-scsi** (SCSI 設備的預設設置)。如果可以的話，**IDE** 設備不推薦用於大多數場景。為了將 **virtio** 與 **Windows**一起使用，必須安裝 **virtio** 驅動程式，**Nutanix** 移動性驅動程式或 **Nutanix VM tools**,其中現代 **Linux** 發行版本預裝了 **virtio**。

```
<controller type='scsi' index='0' model='virtio-scsi'>
```

```

<driver max_sectors='2048'>

<alias name='scsi0'>

<address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'>

</controller>
...

```

當一個活動的 Stargate 下線後(通過 NOP OUT 命令獲取狀態), iSCSI Redirector 將本地的 Stargate 標記為不健康狀態。當 QEMU 試圖發起一個 iSCSI 請求時，Redirector 將重定向此登錄請求到一個健康的 Stargate 之上。

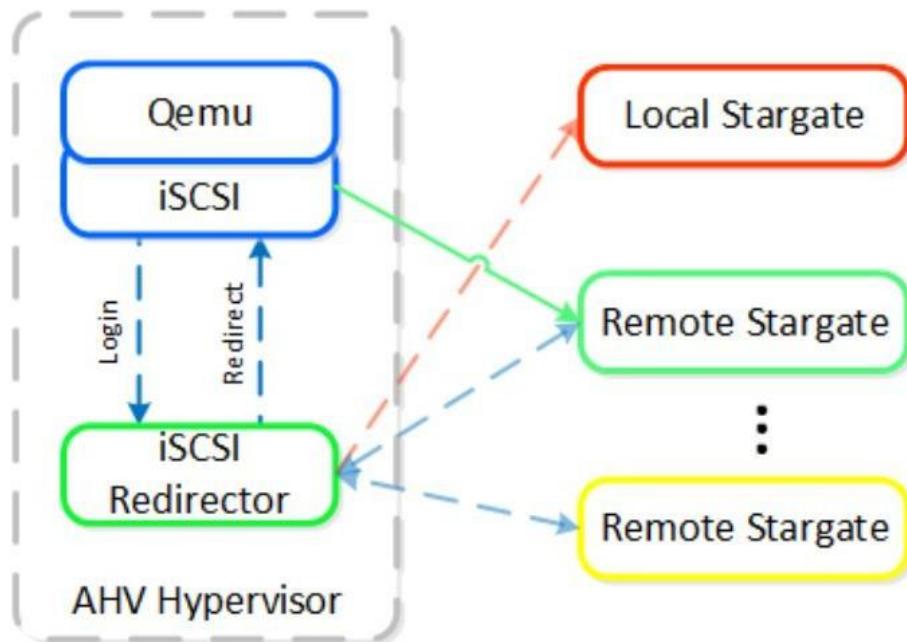


圖. iSCSI 多路徑 – 本地 CVM 下線

一旦本地 CVM 重新上線（通過 NOP OUT 命令獲得狀態），遠端的 Stargate 將結束所有的遠端 iSCSI 會話。QEMU 將再次發送一個新的 iSCSI 登錄請求並重定向到本地的 Stargate。

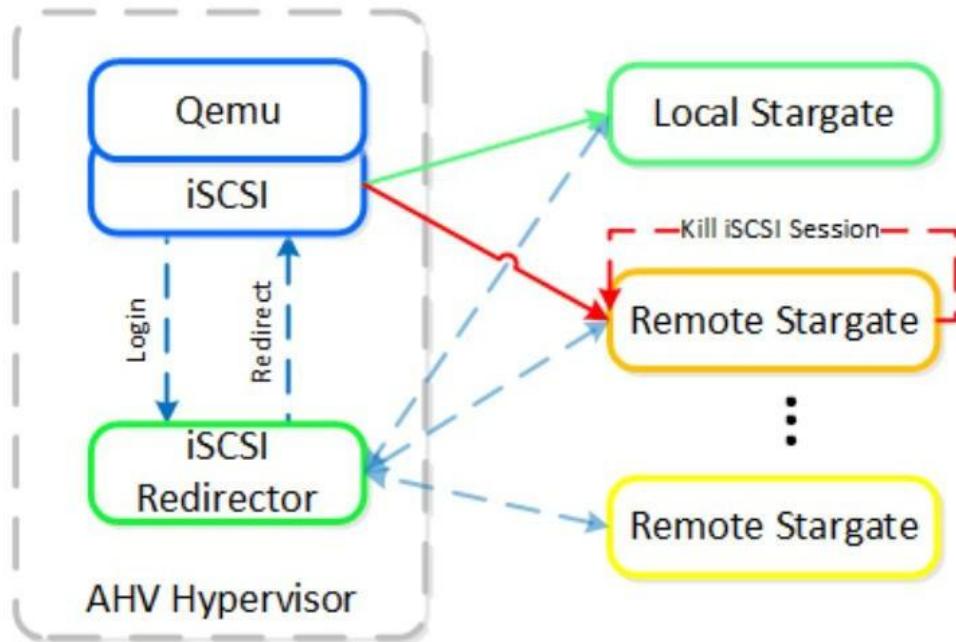


圖. iSCSI 多路徑 – 本地 CVM 上線

5.2.1.1 傳統 I/O 路徑

與每個系統管理程式和作業系統一樣，使用者和內核空間元件混合在一起，它們相互作用以執行一個共同的活動。在閱讀下面的文章之前，建議閱讀“用戶與內核空間”一節，以瞭解它們之間的相互作用。

當一個虛機執行 I/O 時，它將執行以下操作(為清晰起見，一些步驟已經被排除):

1. 虛機的作業系統對虛擬裝置執行 SCSI 命令
2. Virtio-scsi 接受這些請求並將它們放在客戶的記憶體中
3. 請求由 QEMU 主迴圈處理
4. Libiscsi 檢查每個請求並轉發
5. 網路層將請求轉發到本地 CVM(如果本地不可用，則從外部轉發)
6. Stargate 進程組處理請求

下面顯示了這個示例流程:

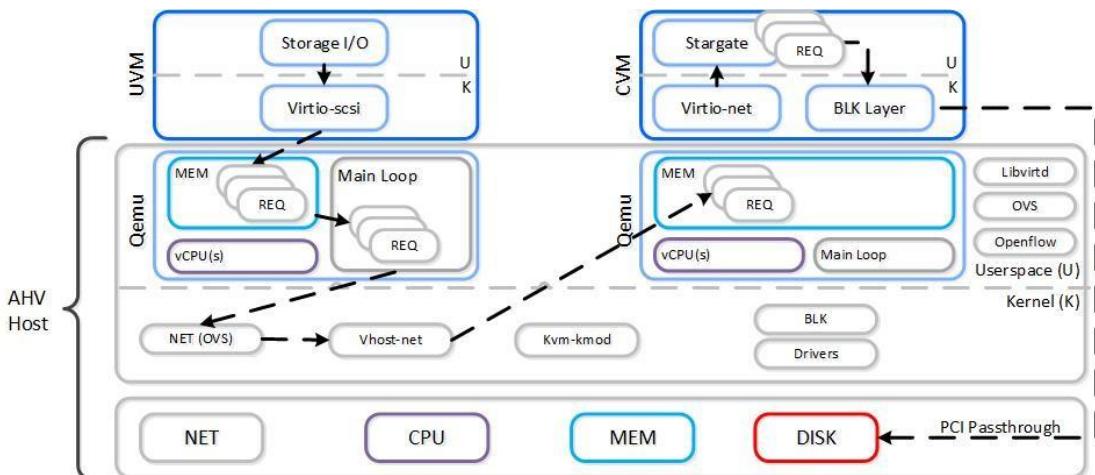


圖. AHV VirtIO 資料路徑 – 經典

看看 Domain 的 XML，我們可以看到它是使用 qemu-kvm 模擬器：

```

...
<devices>
<emulator>/usr/libexec/qemu-kvm</emulator>
...

```

查看 AHV 主機，您可以看到 qemu-kvm 已經使用本地 Bridge 和 ip 建立了與正常的 Stargate 的會話。對於外部通信，將使用外部主機和 Stargate IP 組。注意：每個磁片設備將有一個會話(參見 PID 24845)

```

[root@NTNX-BEAST-1 log]# netstat -np | egrep tcp.*qemu
Proto ... Local Address      Foreign Address      State      PID/Program name
tcp  ... 192.168.5.1:50410  192.168.5.254:3261  ESTABLISHED 25293/qemu-kvm
tcp  ... 192.168.5.1:50434  192.168.5.254:3261  ESTABLISHED 23198/qemu-kvm
tcp  ... 192.168.5.1:50464  192.168.5.254:3261  ESTABLISHED 24845/qemu-kvm

```



```
tcp ... 192.168.5.1:50465 192.168.5.254:3261 ESTABLISHED 24845/qemu-kvm
```

```
...
```

現在在這個路徑中有一些低效的地方，因為主迴圈是單執行緒的，而 `libiscsi` 會檢查每個 SCSI 命令。

5.2.1.2 Frodo I / O 路徑（又名 AHV Turbo 模式）

隨著存儲技術的不斷發展和效率的提高，我們可以進一步優化。鑑於我們完全控制 AHV 和 Nutanix 堆疊，這是一個可以進一步提升的部分。

簡而言之，Frodo 是 AHV 的一個高度優化的 I/O 路徑，它允許更高的輸送量、更低的延遲和更少的 CPU 開銷。

專家提示

Frodo 功能會在 AOS 5.5.X 之後版本的虛擬機器中被啟用

當一個 VM 執行 I/O 時，它將執行以下操作(為清晰起見，一些步驟已經被排除):

1. VM 的作業系統對虛擬裝置執行 SCSI 命令
2. Virtio-scsi 接受這些請求並將它們放在客戶的記憶體中
3. 請求由 Frodo 處理
4. 自訂 libiscsi 附加 iscsi 頭和轉發
5. 網路層將請求轉發到本地 CVM(如果本地不可用，則從外部轉發)
6. Stargate 進程（組）處理請求

下圖顯示了這個示例流程:

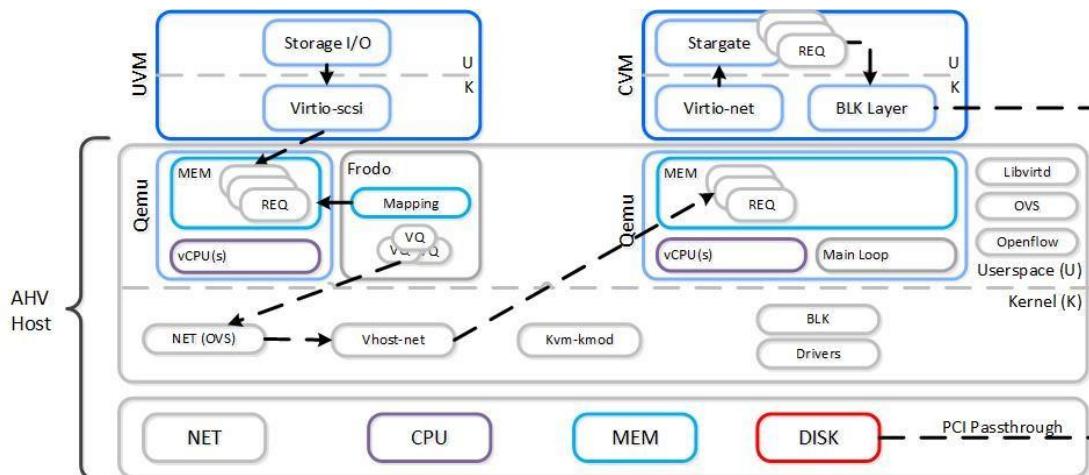


圖. AHV VirtIO 資料路徑- 經典

下面的路徑看起來與傳統的 I/O 很相似，除了一些關鍵的區別：

- qemu 主迴圈被 Frodo (vhost-user-scsi)取代
- frodo 向客戶公開多個虛擬佇列(VQs)(每個 vCPU 一個)
- 為多 vcpu 的虛擬機器使用多個執行緒
- libiscsi 被我們自己的更羽量級的版本所取代

對於用戶端，它將注意到現在有多個用於磁片設備的佇列，而不是只看到性能改進。在某些情況下，與 Qemu 相比，執行 I/O 會減少 25%的 CPU 開銷，並且性能會提高 3 倍!與另一個 hypervisor 相比，我們已經看到執行 I/Os 的 CPU 開銷下降了 3 倍。

看看一個 AHV 主機，你會看到一個 frodo 進程為每個虛機 (qemu-kvm 進程)運行：

```
[root@drt-itppc03-1 ~]# ps aux | egrep frodo
... /usr/libexec/qemu-kvm ... -chardev socket,id=frodo0,fd=3 \
-device vhost-user-scsi-pci,chardev=frodo0,num_queues=16...
```



```
... /usr/libexec/frodo ... 127.0.0.1:3261 -t iqn.2010-06.com.nutanix:vmdisk...
```

```
...
```

看看域的 XML，我們可以看到它是用 **frodo** 模式：

```
<devices>  
  
<emulator>/usr/libexec/frodo</emulator>
```

```
...
```

專家提示

- 要利用 **Frodo** 的多執行緒/連接，必須在啟用時讓虛機 **CPU>= 2 vCPU**。
- 其特徵如下：
 - 1 vCPU 用戶虛機:
 - 1 Frodo 進程/ 會話每一個磁片設備
 - >= 2 vCPU 用戶虛機:
 - 2 Frodo 進程/ 會話每一個磁片設備

在接下來的視頻中，你可以看到 **Frodo** 進程已經建立了一個使用本地 **Bridge** 和 **IP** 的正常的 **Stargate** 會話。對於外部通信，將使用外部主機和 **Stargate IP** 位址。

```
[root@NTNX-BEAST-1 log]# netstat -np | egrep 'tcp.*frodo'  
  
Proto ... Local Address      Foreign Address      State      PID/Program name  
  
tcp  ... 192.168.5.1:39568  192.168.5.254:3261  ESTABLISHED 42957/frodo  
  
tcp  ... 192.168.5.1:39538  192.168.5.254:3261  ESTABLISHED 42957/frodo  
  
tcp  ... 192.168.5.1:39580  192.168.5.254:3261  ESTABLISHED 42957/frodo
```

```
tcp ... 192.168.5.1:39592 192.168.5.254:3261 ESTABLISHED 42957/frod
```

5.2.2 IP 地址管理

Acropolis IP 地址管理(IPAM) 解決方案提供了 DHCP 段以及分配 IP 地址到虛擬機器的能力。IPAM 利用了 VXLAN 和 OpenFlow 規則去中斷和回應 DHCP 請求。

這裡我們通過一個示例來展示通過本地 Acropolis master 上 Nutanix IPAM 方案的 DHCP 請求：

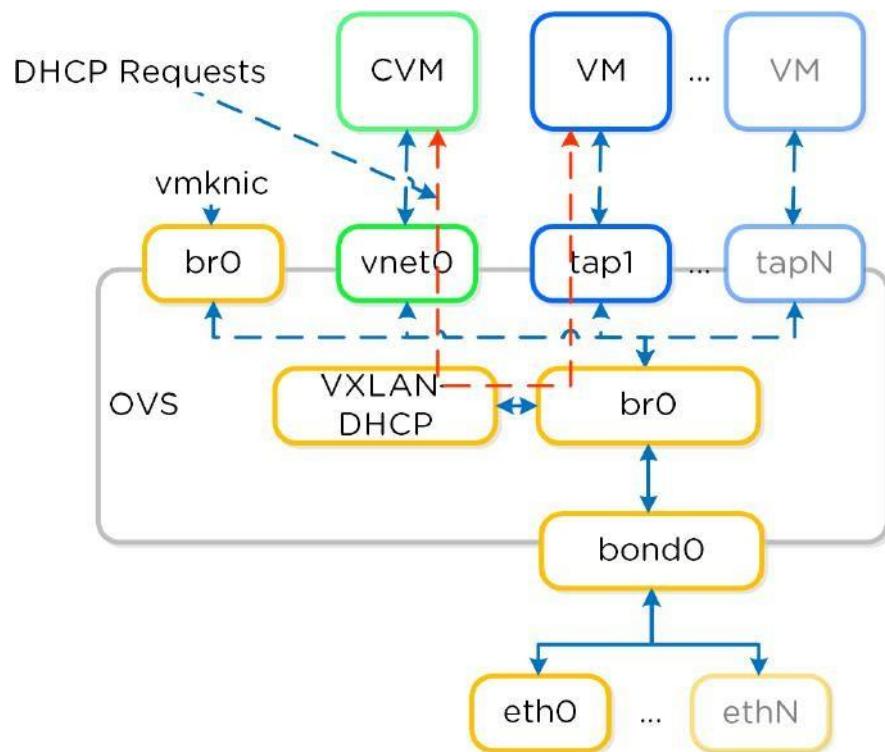


圖 . IPAM – 本地 Acropolis Master

如果一個 Acropolis Master 是運行在遠端的，同一個 VXLAN 隧道啟用並處理網路請求。

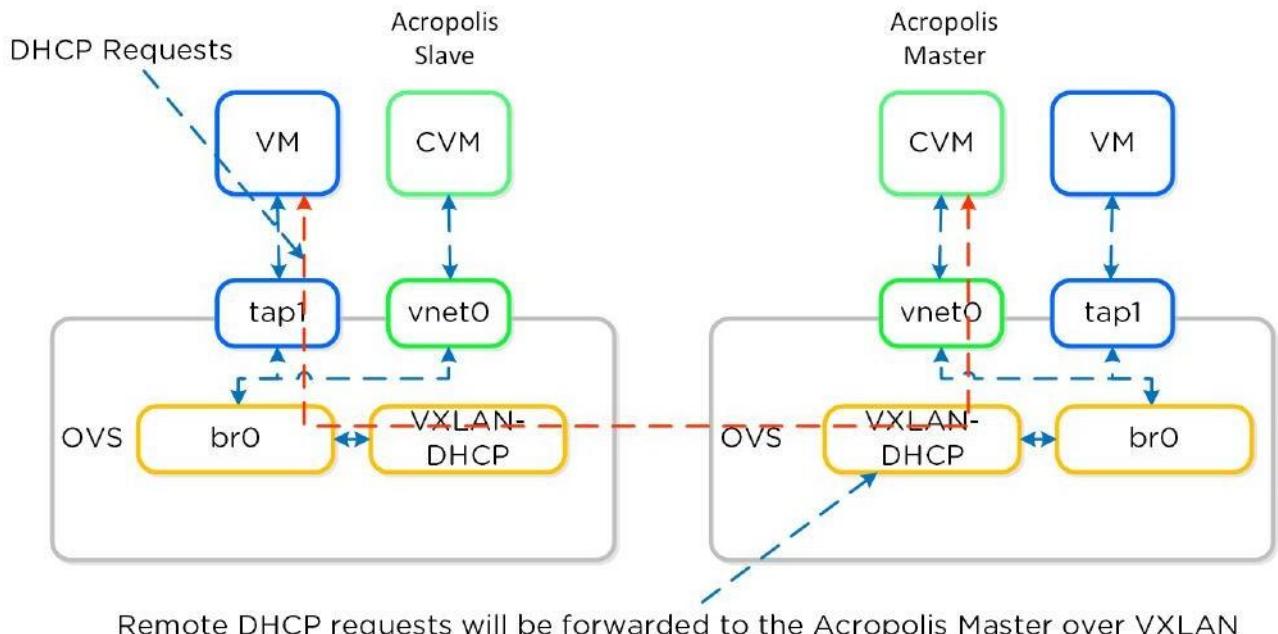


圖. IPAM – 遠程 Acropolis Master

當然，傳統的 DHCP / IPAM 方案也能夠被用於“未管理”的網路場景之中。

5.2.3 VM 高可用性 (HA)

AHV 虛擬機器高可用性 (HA) 是一個用於在主機或機架在出現故障時確保虛擬機器高可用性的功能。當主機出現故障時，虛擬機器將自動在集群中一個健康的節點上重新啟動。Acropolis Master 負責重新啟動虛擬機器到健康的節點之上。

Acropolis Master 通過監控集群中所有主機上的 libvirt 來追蹤主機的健康狀態：

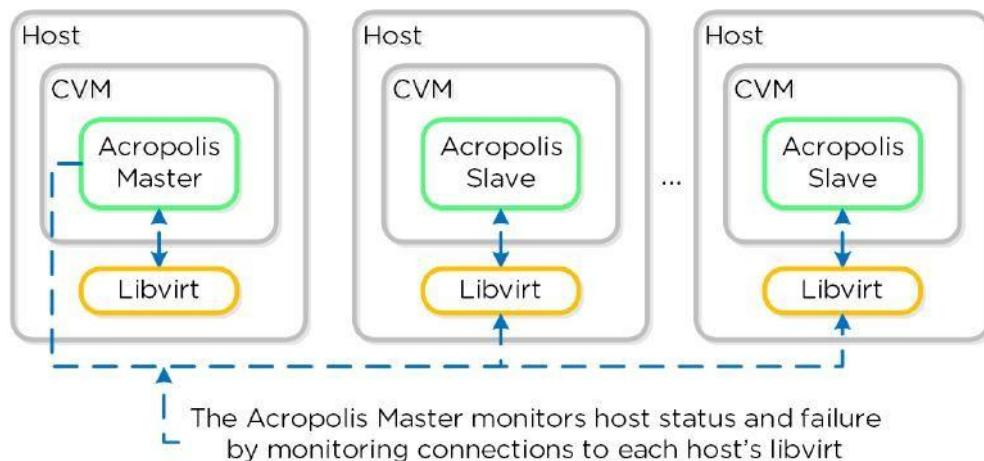


圖. HA – 主機監控



當 Acropolis Master 出現分區、隔離或者失敗時，一個新的 Acropolis Master 將自動在集群中的健康節點中選舉出來。如果一個集群出現分區的情況（舉例來說，X 節點不能與其他的 Y 節點通迅）時，仲裁側的主機將保持線上，同時所有的虛擬機器將在這些主機上重新開機。

以下是兩個主要的 HA 資源預留類型：

- 缺省
 - 該模式不需要配置，安裝基於 AHV 的集群時卻生設置。當 AHV 主機不可用時，故障主機上的虛擬機會根據可用資源重啟在剩下的主機，如果剩餘主機沒有足夠資源，不是所有受影響的虛擬機會重啟。
- 保證
 - 非缺省配置，當主機故障時，在集群 AHV 主機上預留空間確保所有受故障影響的虛擬機器都能重啟在 AHV 集群的其它主機上。為了啟用保證模式，選擇啟用 HA 選擇框。會提示預留的記憶體數量和可以容忍的 AHV 主機故障數量。

5.2.3.1 資源預留

當使用保證模式，系統會預留主機資源。預留數量如下：

- 如果所有 container 時 RF2 (FT1)
 - 相當於一個主機的資源
- 如果任何 container 時 RF3 (FT2)
 - 相當於兩個主機的資源
-

當主機記憶體容量不相等，系統使用最大主機記憶體決定每個主機預留多少資源。

預留分段會將資源預留在集群的所有主機上。這種場景下，每個主機為 HA 預留一個百分比，確保整個集群在主機故障時有足夠的故障切換的空間重啟虛擬機器。

下圖顯示了一個預留分段的示例場景：

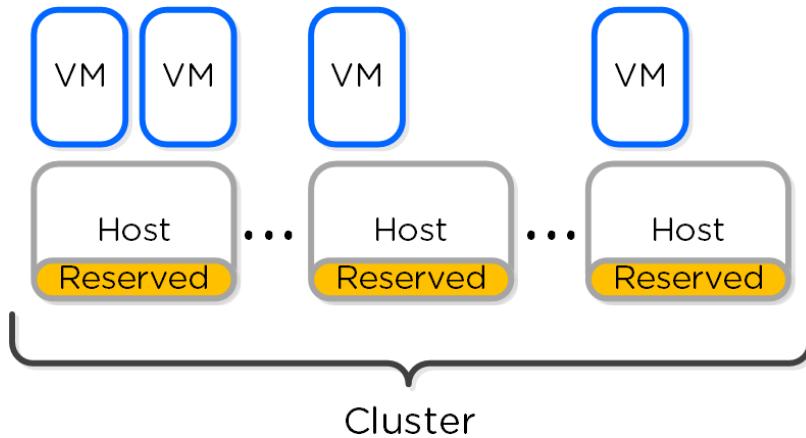


圖. HA – 預留分段

當一個主機失效時，其上運行的所有虛擬機器將在集群中剩餘的健康主機上重新開機。

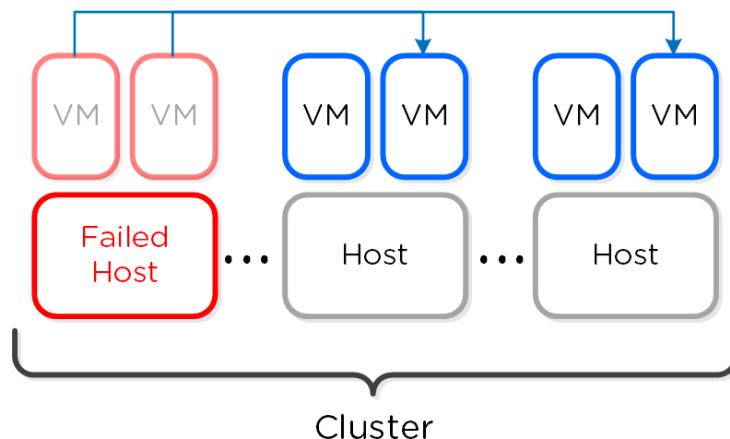


圖. HA – 預留分段- 容錯移轉

預留分段計算

系統將自動計算總共的分段數量和每個主機上的預留值。下面我們將描述關於計算分段預留的一些細節。

Acropolis HA 使用固定分段去預留當主機失效時可以成功重新開機虛擬機器的空間。這個分段容量與系統中最大記憶體配置的虛擬機器的記憶體一致。Acropolis HA 這個特色功能提供了組合多個小容量虛擬機器到一個單一的固定容量分段的能力。在具有不同虛擬機器容量的集群中，單一分段能夠容納多個虛擬機器，因此可以減少固定容量的分段的碎片化。

最有效的虛擬機器放置方式(最小數量的分段預留)被定義成一個裝箱問題，它是一個著名的電腦科學問題。最佳的解決方案是使用非確定性多項式（指數-科學術語），但是探索型的方案可能更適用於大部分通用場景。Nutanix 將繼續提高虛擬機器放置演算法。我們期待在未來的版本中，通用場景僅有 0.25 的開銷，當前碎片的開銷在 0.5 到 1 之間。每個配置的失效主機的總體開銷在 1.5-2 之間。

當使用一個分段式的預留時，以下是一些關鍵的資料：



- 分段容量= 最大的虛擬機器的記憶體(GB)
- 最高負載的主機=最多運行虛擬機器記憶體的主機(GB)
- 碎片開銷= 0.5 - 1

基於上述的輸入，你能計算出預留分段的數量：

預留分段= (最高負載的主機/ 分段容量) x (1 + 碎片開銷)

5.3 管理

5.3.1 重要頁*

5.3.2 命令參考

在 OVS 上僅啟動 10GbE 鏈路

說明: 在本機的 bond0 上將啟用 10g 鏈路

```
manage_ovs --interfaces 10g update_uplinks
```

說明: 在整個集群上僅啟用 10g 鏈路

```
allssh "manage_ovs --interfaces 10g update_uplinks"
```

顯示 OVS 上聯鏈路

說明: 在本機上顯示 ovs 上聯鏈路情況

```
manage_ovs show_uplinks
```

說明: 在整個集群上顯示 ovs 上聯鏈路情況

```
allssh "manage_ovs show_uplinks"
```

顯示 OVS 介面



說明: 顯示本機的 ovs 介面

```
manage_ovs show_interfaces
```

說明: 顯示整個集群的 ovs 介面

```
allssh "manage_ovs show_interfaces"
```

顯示 OVS 交換機資訊

說明: 顯示交換機資訊

```
ovs-vsctl show
```

列出 OVS 橋

說明: 列出橋

```
ovs-vsctl list br
```

顯示 OVS 埠資訊

說明: 顯示 OVS 埠資訊

```
ovs-vsctl list port br0
```

```
ovs-vsctl list port <bond>
```

顯示 OVS 介面資訊

說明: 顯示介面資訊

```
ovs-vsctl list interface br0
```

在橋上顯示埠/介面

說明: 顯示上的埠

```
ovs-vsctl list-ports br0
```

說明: 顯示橋上的 iface

```
ovs-vsctl list-ifaces br0
```



創建 OVS 橋

說明: 創建橋

```
ovs-vsctl add-br <bridge>
```

為橋添加埠

說明: 為橋添加埠

```
ovs-vsctl add-port <bridge> <port>
```

說明: 為橋添加 bond 埠

```
ovs-vsctl add-bond <bridge> <port> <iface>
```

顯示 OVS bond 資訊

說明: 顯示 bond 資訊

```
ovs-appctl bond/show <bond>
```

示例:

```
ovs-appctl bond/show bond0
```

設置 bond 模式並配置 LACP

說明: 在埠上啟用 LACP

```
ovs-vsctl set port <bond> lacp=<active/passive>
```

說明: 啟用所有主機上的 bond0

```
for i in `hostips` ;do echo $i; ssh $i source /etc/profile > /dev/null 2>&1; ovs-vsctl set port bond0
```

```
lacp=active;done
```

顯示 bond 上的 LACP 資訊

說明: 顯示 LACP 細節



```
ovs-appctl lacp/show <bond>
```

設置 **bond** 模式

說明: 設置埠的 **bond** 模式

```
ovs-vsctl set port <bond> bond_mode=<active-backup, balance-slb, balance-tcp>
```

顯示 **OpenFlow** 資訊

說明: 顯示 OVS openflow 細節

```
ovs-ofctl show br0
```

說明: Show OpenFlow rules

```
ovs-ofctl dump-flows br0
```

得到 **QEMU** 進程 ID 和 **top** 資訊

說明: 得到 QEMU 的進程 ID 資訊

```
ps aux | grep qemu | awk '{print $2}'
```

說明: 得到指定進程 ID 的性能計量資訊

```
top -p <PID>
```

得到所有 **QEMU** 進程的活動 **Stargate**

說明: 為所有 QEMU 進程得到活動的 Stargate 資訊

```
netstat -np | egrep tcp.*qemu
```

5.3.3 計量與閾值

5.3.4 故障處理& 高級管理

檢查 **iSCSI Redirector** 日誌

說明: 檢查所有主機的 iSCSI Redirector 日誌



```
for i in `hostips` ; do echo $i; ssh root@$i cat /var/log/iscsi_redirector;done
```

單個主機示例

```
Ssh root@<HOST IP>
```

```
Cat /var/log/iscsi_redirector
```

監控 CPU 閑置(閒置 CPU)

說明: 監控 CPU 閑置情況

執行 top 並查看 %st (如下)

```
Cpu(s): 0.0%us, 0.0%sy, 0.0%ni, 96.4%id, 0.0%wa, 0.0%hi, 0.1%si, 0.0%st
```

監控虛擬機器網路資源狀態

說明: 監控虛擬機器資源狀態

執行 virt-top

```
Virt-top
```

查看網路頁，可以按數位 2 進入網路頁面。

6 第六部分：vSphere

6.1 架構

6.1.1 節點架構

在 ESXi 的部署中，控制器虛擬機器（CVM）硬碟使用 VMDirectPath I/O 方式。這使得完整的 PCI 控制器（和附加設備）通過直通方式連接 CVM 並繞過虛擬化層。

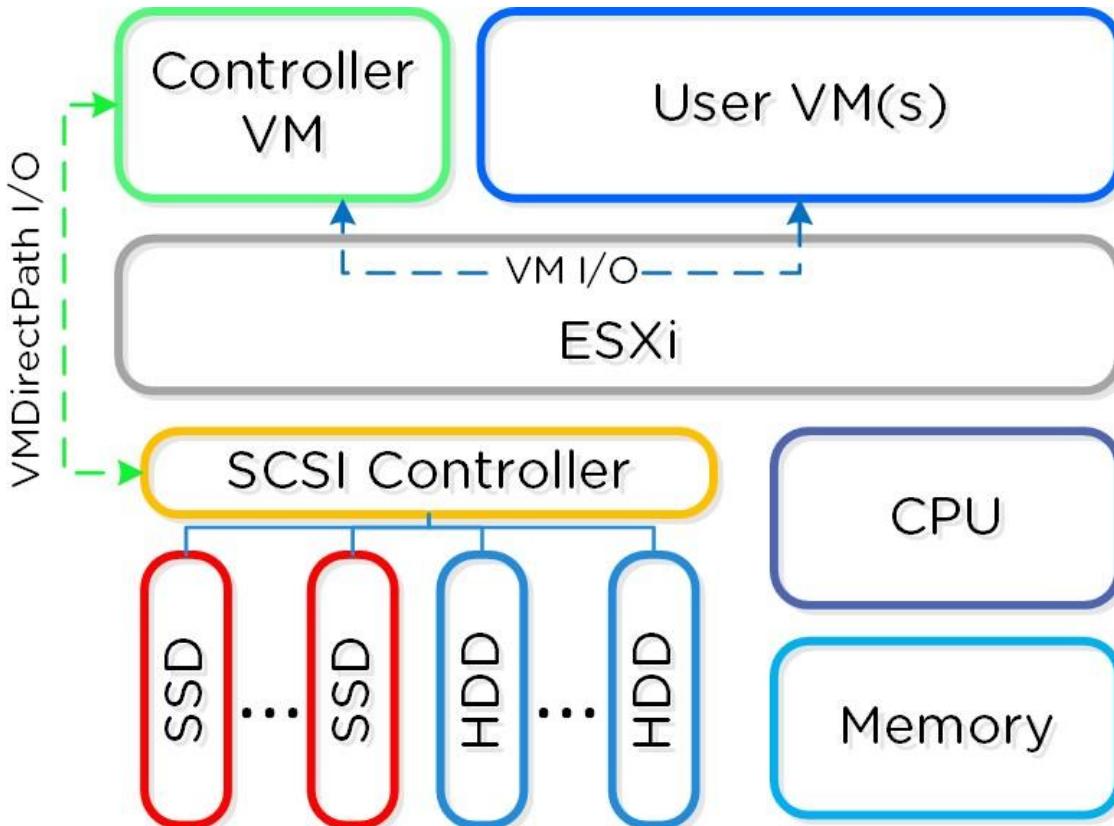


圖. ESXi 的節點架構

6.1.2 最高配置和可擴展性

下面的最高配置和可擴展性的限制適用：

- 最大集群大小：**64**
- 每個虛擬機器的最大虛擬 **CPU : 128**
- 每個虛擬機器最大記憶體：**4TB**
- 最大虛擬磁片容量：**62TB**
- 每台主機最大的虛擬機器：**1024**
- 每個集群最大的虛擬機器：**8000** (如果啟用 **HA 2048** 每個資料存儲)

注：適用於 vSphere 6.0

專家提示

在 ESXi 主機執行基準測試時，必須將測試 ESXi 主機的電源策略設置為“高性能”。這會禁用 P- 和 C- 狀態，保證測試結果沒有人為限制。

6.1.3 網路



每個 ESXi 主機有一個本地的 vSwitch 用來做 Nutanix CVM 和主機之間的內部通信。對於虛擬機器和外部的通信則利用標準的 vSwitch (默認) 和 dvSwitch 來實現。

本地 vSwitch (vSwitchNutanix) 用來做 Nutanix CVM 和 ESXi 主機的本地通信。主機上有 vmkernel 埠與 vSwitch (vmk1 - 192.168.5.1) 相連，CVM 有一個介面綁定在內部 switch 的埠組上 (svm-iscsi-pg - 192.168.5.2)。這是一個私有的存儲通信路徑。

外部 vSwitch 可以是標準的 vSwitch 或者 dvSwitch，主要負責 ESXi 主機和 CVM 的外部介面以及虛擬機器在主機上所需要的埠組。外部 vmkernel 介面用來做主機管理、vMotion 等。外部 CVM 介面用來做和其他 Nutanix CVM 的通信。如果 Trunk 上開啟 VLAN，那麼埠組就可以根據需要創建。

下圖顯示了 vSwitch 的結構：

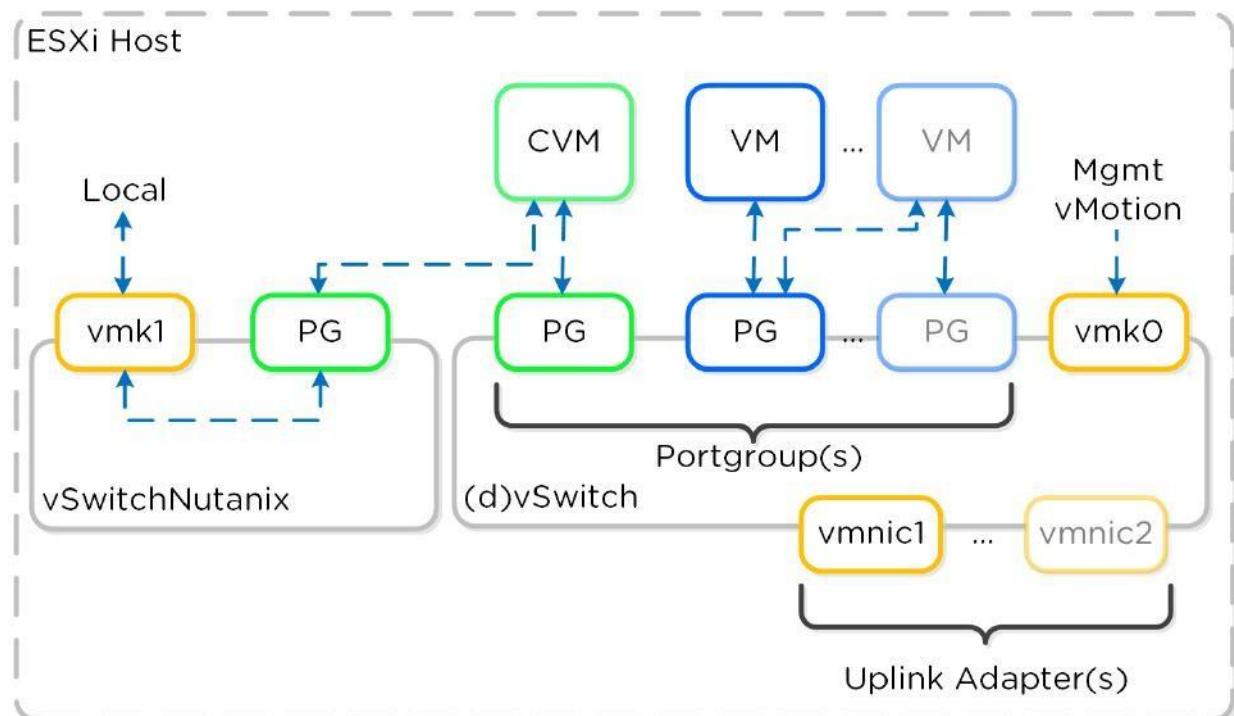


圖. ESXi vSwitch 網路概述

上行和鏈路綁定策略



推薦採用兩個 ToR 交換機和上行鏈路 來保證交換機的 HA。系統預設的上行介面是 active/passive 模式。具有 active/active 上行介面(e.g. vPC, MLAG, etc.)的上行交換機可以用作其他的網路流量。

6.2 如何工作

6.2.1 磁碟陣列卸載負載—VAAI

Nutanix 平臺支持 VMware API 進行陣列集成（VAAI），它允許 hypervisor 卸載某些任務負載到陣列。這將使 Hypervisor 效率更高並且不需要“中間人”。Nutanix 目前支持 NAS 的 VAAI 原語，包括“全檔克隆”，“快速檔克隆”和“預留空間”原語。這裡有一個很好的文章，解釋的各種原語：<http://cormachogan.com/2012/11/08/vaai-comparison-block-versus-nas/>。

- 克隆虛擬機器有快照 -> VAAI 將不能使用
- 克隆虛擬機器沒有快照且處於關機狀態 -> VAAI 將被使用
- 克隆虛擬機器到不同的資料存儲/容器 -> VAAI 將不能使用
- 克隆開機狀態的虛擬機器 -> VAAI 將不能使

用這些方案適用於 VMware View :

- View 完整克隆（範本有快照）-> VAAI 將不能使用
- View 完整克隆（範本沒有快照）-> VAAI 將被使用
- View 連結克隆（VCAI）-> VAAI 將被使用

您可以驗證 VAAI 的操作，通過使用“NFS 適配器”活動跟蹤頁面。

6.2.2 CVM Autopathing aka Ha.py

在本節中，我將介紹 CVM 的“失敗”的處理方式。一個 CVM“失敗”可能包括用戶將 CVM 關機， CVM 升級，或任何可能會搞垮 CVM 的事件。DSF 有一個稱為 autopathing 特徵，其中當一個本地 CVM 變得不可用時，I/O 透明傳遞給集群中的其他 CVM 處理。Hypervisor 和 CVM 通信使用一個專用的 vSwitch 的私網位址 192.168.5.0 (詳見上圖)。這意味著，對於所有存儲的 I/O，這些都發生在 CVM 的內部 IP 位址 (192.168.5.2) 上。CVM 的外部 IP 位址用於遠端複製和 CVM 通信。下圖顯示了這樣的一個例子：

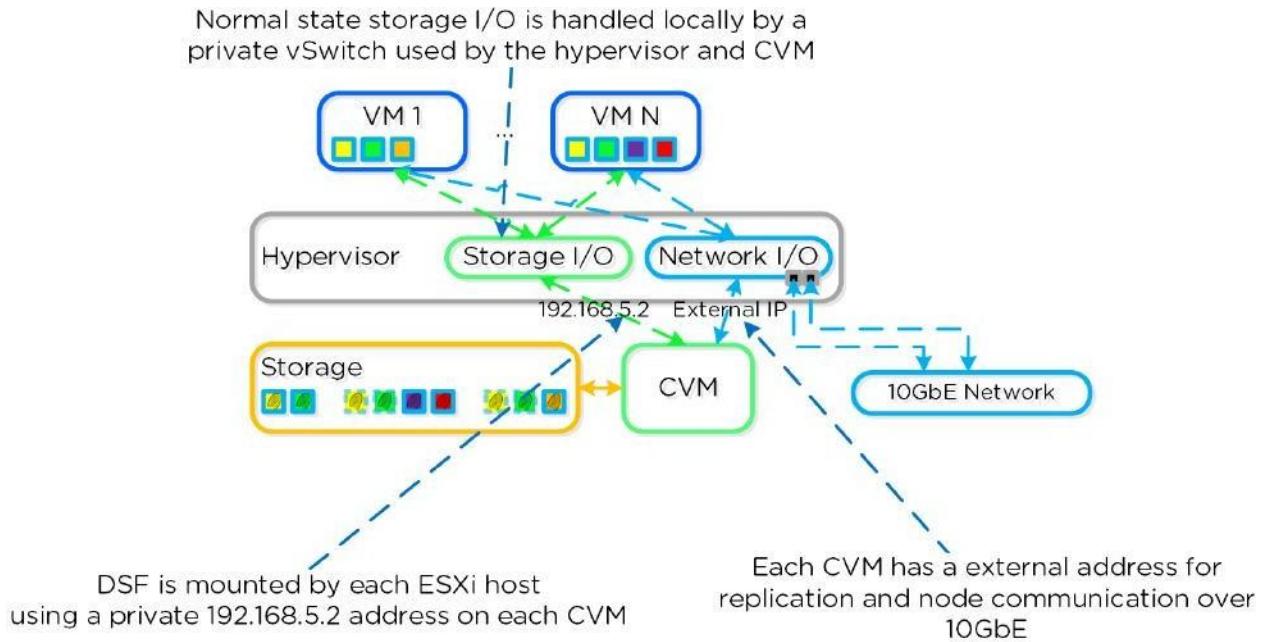


圖. ESXi 主機網路

在一個本地的 CVM 故障的情況下，先前由本地 CVM 託管在本地 192.168.5.2 地址是不可用的。DFS 將自動檢測到該故障並通過萬兆網路重定向這些 I/O 到集群中另一個 CVM 上。Hypervisor 和主機上運行的虛擬機器將以透明的方式重新路由。這意味著，即使一個 CVM 關機，虛擬機器仍然會繼續能夠執行 I/O。一旦本地 CVM 的還原並可用，流量隨後將無縫地轉交給本地 CVM 繼續服務。

下圖顯示了如何尋找一個失敗的 CVM 的圖示：

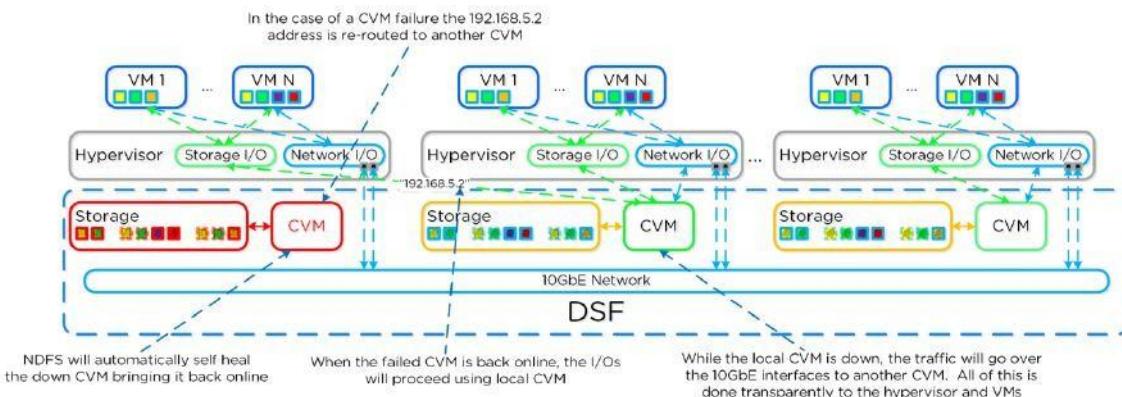


圖 . ESXi 主機網路 - 本地 CVM 下線

6.3 管理

6.3.1 重要頁*

6.3.2 命令參考



ESXi 的集群升級

說明：使用 CLI 執行 ESXi 主機的自動升級

```
#上傳離線升級包到 Nutanix NFS 容器
```

```
#登錄到 Nutanix CVM
```

```
#進行升級
```

```
cluster --md5sum=<bundle_checksum> --bundle=</path/to/offline_bundle> host_upgrade
```

#示例

```
cluster --md5sum=bff0b5558ad226ad395f6a4dc2b28597 --bundle=/tmp/VMware-ESXi-5.5.0-1331820-depot.zip host_upgrade
```

重新開機 ESXi 主機服務

說明：以一個增量的方式重新開機每個 ESXi 主機服務

```
for i in `hostips`;do ssh root@$i "services.sh restart";done
```

顯示 ESXi 主機的網卡“up”狀態

說明：顯示 ESXi 主機的在“UP”狀態的網卡

```
for i in `hostips`;do echo $i && ssh root@$i esxcfg-nics -l | grep Up;done
```

顯示 ESXi 主機的 10GbE 網卡和狀態

說明：顯示 ESXi 主機的 10GbE 網卡和狀態

```
for i in `hostips`;do echo $i && ssh root@$i esxcfg-nics -l | grep ixgbe;done
```

顯示 ESXi 主機的活動適配器

說明：顯示 ESXi 主機的工作，待機和未使用的適配器

```
for i in `hostips`;do echo $i && ssh root@$i "esxcli network vswitch standard policy failover get --vswitch-name vSwitch0";done
```

顯示 ESXi 主機路由表

說明：顯示 ESXi 主機的路由表

```
for i in `hostips`;do ssh root@$i 'esxcfg-route -l';done
```

檢查 VAAI 在 Datastore 上是否啟用

說明：檢查 VAAI 在 Datastore 上是否啟用/支援

```
vmkfstools -Ph /vmfs/volumes/<Datastore Name>
```



設定 VIB 校驗級別為 community supported

說明：設置 VIB 校驗程度為 CommunitySupported 允許協力廠商安裝 VIB

```
esxcli software acceptance set --level CommunitySupported
```

安裝 VIB

說明：不檢查簽名安裝 VIB

```
esxcli software vib install --viburl=<VIB directory>/<VIB name> --no-sig-check
```

或者

```
esxcli software vib install --depoturl=<VIB directory>/<VIB name> --no-sig-check
```

檢查 ESXi 的 ramdisk 的可用空間

說明：檢查 ESXi ramdisk 可用空間

```
for i in `hostips`;do echo $i; ssh root@$i 'vdf -h';done
```

清除 pynfs 日誌

說明：清除每個 ESXi 主機上的 pynfs 記錄

```
i in `hostips`;do echo $i; ssh root@$i '> /pynfs/pynfs.log';
```

6.3.3 指標和閾值 *

6.3.4 故障排除和高級管理*

7 第七部分：Hyper-V

7.1 架構

當 Nutanix Hyper-V 集群創建時，會自動將 Hyper-V 主機加入指定的 Windows AD 域。然後加入為虛擬機器 HA 創建的 failover 集群。完成後，每個獨立的 Hyper-V 主機和 failover 集群都會有一個 AD 物件。

7.1.1 節點架構

在 Hyper-V 部署中，控制器虛擬機器（CVM）作為一個虛擬機器運行，並使用磁片直通方式（passthrough）管理磁片。

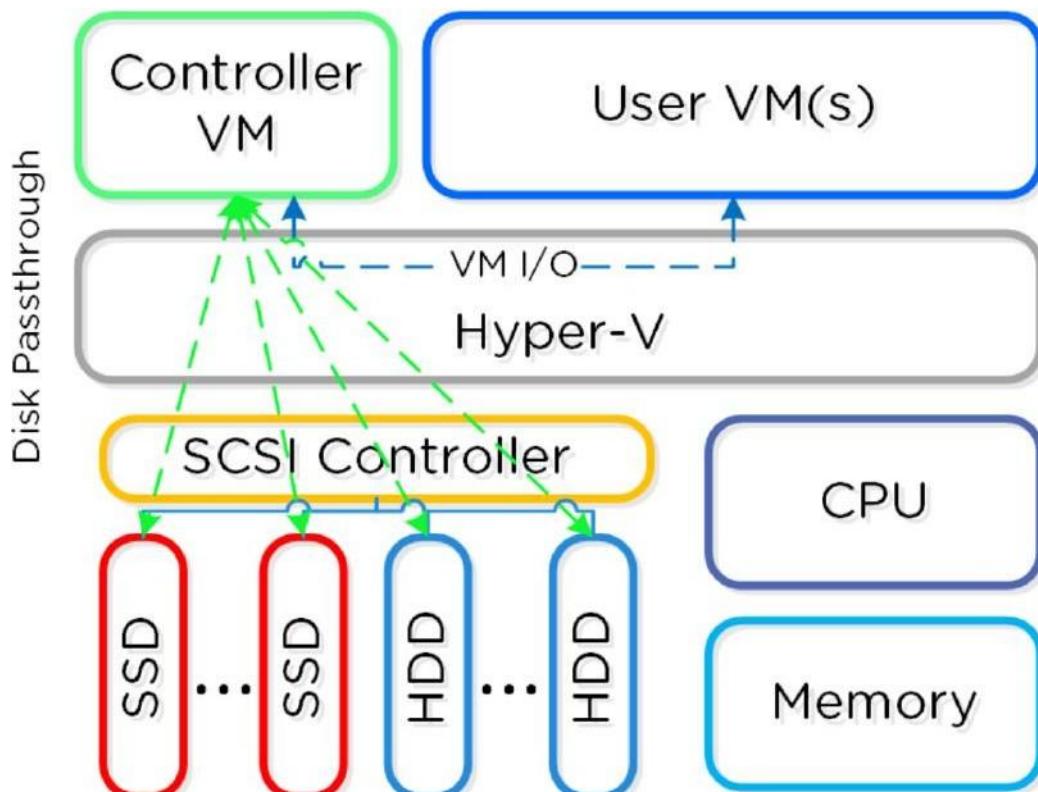


圖. Hyper-V 的節點架構

7.1.2 最高配置和可擴展性

下麵是最高配置和可擴展性的限制：

- 最大集群大小：**64**
- 每個 VM 最大的 **vCPU**：**64**
- 每個虛擬機器最大記憶體：**1TB**
- 最大虛擬磁片容量：**64TB**
- 每台主機最大的虛擬機器：**1024**
- 每個集群最大的虛擬機器：

8000注：適用 Hyper-V 2012 R2 版本

7.1.3 網路

每個 Hyper-V 主機有一個內部的虛擬交換機用來做 Nutanix CVM 和主機之間的內部通信。對於虛擬機器和外部的通信則利用外部的虛擬交換機（預設）或邏輯交換機來實現。

內部交換機 (InternalSwitch)用來做 Nutanix CVM 和 Hyper-V 主機的本地通信。主機上有 vmkernel 埸與 vSwitch (vmk1 - 192.168.5.1)相連， CVM 有一個介面綁定在內部 switch 的埠組上 (svm-iscsi-pg - 192.168.5.2)。在這個內部交換機上， 主機有一個虛擬乙太網介面 (vEth) (192.168.5.1)，CVM 在內部交換機上有一個虛擬乙太網介面 vEth (192.168.5.2)。這是一個私有的存儲通信路徑。

外部 vSwitch 可以是標準的 vSwitch 或者邏輯交換機，主要負責 Hyper-V 主機和 CVM 的外部介面以及虛擬機器在主機上所需要邏輯網路。外部 vEth 介面用來做主機管理、線上遷移等。外部 CVM 介面用來做和其他 Nutanix CVM 的通信。如果 Trunk 上開啟 VLAN，那麼就可以根據需要創建埠組。

下圖顯示了虛擬交換機的結構：

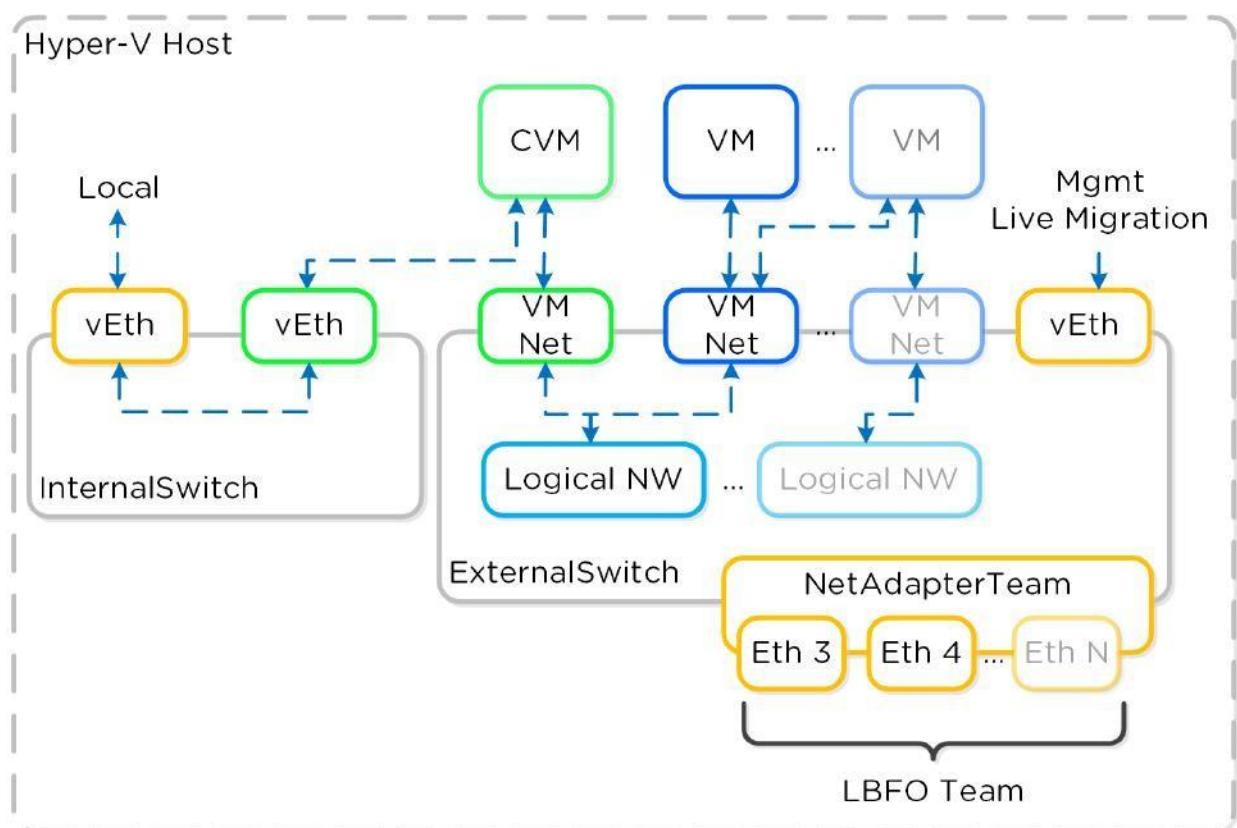


圖. Hyper-V 虛擬交換機網路概覽

上行和鏈路綁定策略



推薦採用兩個 ToR 交換機和上行鏈路來保證交換機的 HA。系統在交換機獨立模式預設採用 LBFO 綁定，這無需其他特殊的配置。

7.2 如何工作

7.2.1 磁碟陣列卸載負載—ODX

Nutanix 平臺支持微軟 Offloaded Data Transfer (ODX)，允許 Hypervisor 卸載某些任務負載到陣列。這將使 Hypervisor 效率更高並且不需要“中間人”。Nutanix 目前為 SMB 支援 ODX，其中包括完整的複製和歸零操作。然而，與 VAAI 中“快速的檔”克隆操作（使用可寫快照）相反，執行完全拷貝。鑑於此，更有效的方法是依靠本地 DSF 克隆，目前可以通過調用通過 nCLI、REST、或 Powershell CMDlets 命令。目前 ODX 由以下操作調用：

- 在 DSF SMB 共用上的虛擬機器或虛擬機器檔副本
- SMB 共用檔副本

從 SCVMM 庫（DSF SMB 共用）部署的範本 - 注：共用必須被添加到使用短名稱的 SCVMM 集群（例如，不是 FQDN）。一個簡單的方法使這個條目添加到主機檔的集群（例如：10.10.10.10 nutanix-130）。

ODX 不能通過以下操作來調用：

- 通過 SCVMM 克隆虛擬機器
- 從 SCVMM 庫（非 DSF SMB 共用）部署範本
- XenDesktop 的克隆部署

通過使用“NFS 適配器”的活動跟蹤頁面（是的 NFS，儘管正在通過 SMB 執行）可以驗證 ODX 操作是否發生。當複製 vDISK 時該操作活動顯示“NfsSlaveVaaiCopyDataOp”當一塊磁片置 0 時顯示“NfsSlaveVaaiWriteZerosOp”。

7.3 管理

7.3.1 重要頁*

7.3.2 命令參考

在多台遠端主機上執行命令



說明：在一個或多個遠端主機執行一個 PowerShell

```
$targetServers = "Host1","Host2","Etc"  
Invoke-Command -ComputerName $targetServers {  
    <COMMAND or SCRIPT BLOCK>  
}
```

檢查可用 VMQ Offloads

說明：顯示特定主機 VMQ Offloads 的可用數量

```
gwmi - Namespace "root\virtualization\v2" - Class MsVm_VirtualEthernetSwitch | select elementname,  
MaxVMQOffloads
```

禁用匹配特定首碼虛擬機器的 VMQ

說明：禁用特定虛擬機器的 VMQ

```
$vmPrefix = "myVMs"  
Get-VM | Where {$_.Name -match $vmPrefix} | Get-VMNetworkAdapter | Set-VMNetworkAdapter -VmqWeight  
0
```

啟用匹配特定首碼虛擬機器的 VMQ

說明：啟用特定虛擬機器的 VMQ

```
$vmPrefix = "myVMs"  
Get-VM | Where {$_.Name -match $vmPrefix} | Get-VMNetworkAdapter | Set-VMNetworkAdapter -VmqWeight  
1
```

開機特定首碼的虛擬機器

說明：將特定首碼的虛擬機器開機

```
$vmPrefix = "myVMs"  
Get-VM | Where {$_.Name -match $vmPrefix -and $_.StatusString -eq "Stopped"} | Start-VM
```

關機特定首碼的虛擬機器

說明：將特定首碼的虛擬機器關機

```
$vmPrefix = "myVMs"  
Get-VM | Where {$_.Name -match $vmPrefix -and $_.StatusString -eq "Running"} | Shutdown-VM -  
RunAsynchronously
```

停止特定首碼的虛擬機器



說明：將特定首碼的虛擬機器停止

```
$vmPrefix = "myVMs"
```

```
Get-VM | Where {$_.Name -match $vmPrefix} | Stop-VM
```

獲取 Hyper-V 主機的 RSS 設置

說明：獲取 Hyper-V 主機 RSS(recvieve side scaling) 設置

```
Get-NetAdapterRss
```

檢查 Winsh 和 WinRM 連接

說明：通過執行一個簡單的查詢檢查 Winsh 和 WinRM 連接/狀態，應返回“the computer system object not an error”

```
allssh "winsh "get-wmiobject win32
```

7.3.3 指標和閾值*

7.3.4 故障排除和高級管理*

8 第八部分：Nutanix 集群

8.1 AWS 上的 Nutanix 群集

AWS 上的 Nutanix 集群提供了使用裸金屬資源的按需供應的集群。通過 Nutanix 平臺的簡單性，可以實現真正的隨需應變能力。供應的集群看起來很像傳統的 AHV 集群，只是運行在雲提供商資料中心而已。

8.1.1 已支援的配置

解決方案適用於以下配置(清單可能不完整，請參考文檔獲得完全支持的列表):

核心使用場景：

- 按需/突發容量
- 備份/ DR



- 原生雲
- GEO 擴展/ 資料中心合併
- 應用程式遷移
- 等等。

管理介面：

- Nutanix 集群門戶 – 資源供應
- Prism Central (PC) – Nutanix 上層管理
- AWS 控制台- AWS 管理

支援環境：

- 雲:
 - AWS (EA)
- EC2 基礎實例類型:
 - i3.metal
 - m5d.metal
 - z1d.metal

更新:

- 部分 AOS

相容特性:

- AOS 特性
- AWS 服務

8.1.2 關鍵術語 / 結構

在本節中使用的主要項目如下:

- **Nutanix Clusters Portal**
Nutanix 集群門戶負責處理集群供應請求，並與 AWS 和所供應的主機進行交互。它創建特定于集群的細節並處理動態雲形成堆疊的創建。
- **Region**
多個可用區域(地點)所在的地理地塊或區域。一個區域可以有兩個或多個 AZs。這些區域可以包括 US-East-1 或 US-West-1。
- **Availability Zone (AZ)**
AZ 由一個或多個離散的資料中心組成，通過低延遲連結相互連接。每個網站都有自己的冗餘電源、冷卻系統、網路等。與傳統的資料中心相比，這些被認為更有



彈性，因為 AZ 可以由多個獨立的資料中心組成。這些可能包括 US-East-1a 或 US-West-1a 這樣的網站。

- **VPC**
租戶的 AWS 雲的邏輯隔離段。提供一種機制來保護和隔離環境。可以暴露給互聯網或其他私有網路段(其他 vpc 或 vpn)。
- **S3**
amazon 的物件服務，提供通過 S3 API 訪問的持久物件存儲。這用於存檔/恢復
- **EBS**
物件導向 amazon 的卷/塊服務，它提供可以附加到 ami 的持久卷。
- **Cloud Formation Template (CFT)**
雲形成範本簡化了供應，同時允許您定義資源和依賴項的“堆疊”。然後可以將這個堆疊作為一個整體供應，而不是每個單獨的資源供應。

8.1.3 架構

從高層次上講，Nutanix 集群門戶是在 AWS 上提供 Nutanix 集群並與 AWS 進行交互的主要介面。

設置過程可以概括為以下高級步驟：

1. 在 Nutanix 群集門戶中創建群集
2. 部署特定的輸入（例如，區域，可用區，實例類型，VPC /子網等）
3. Nutanix Cluster Orchestrator 創建關聯的資源
4. Nutanix AMI 中的主機代理通過 AWS 上的 Nutanix 群集簽入
5. 一旦所有主機都啟動，群集就被創建了

以下內容展示了基於 AWS 交互的 Nutanix 群集的高層級概述：

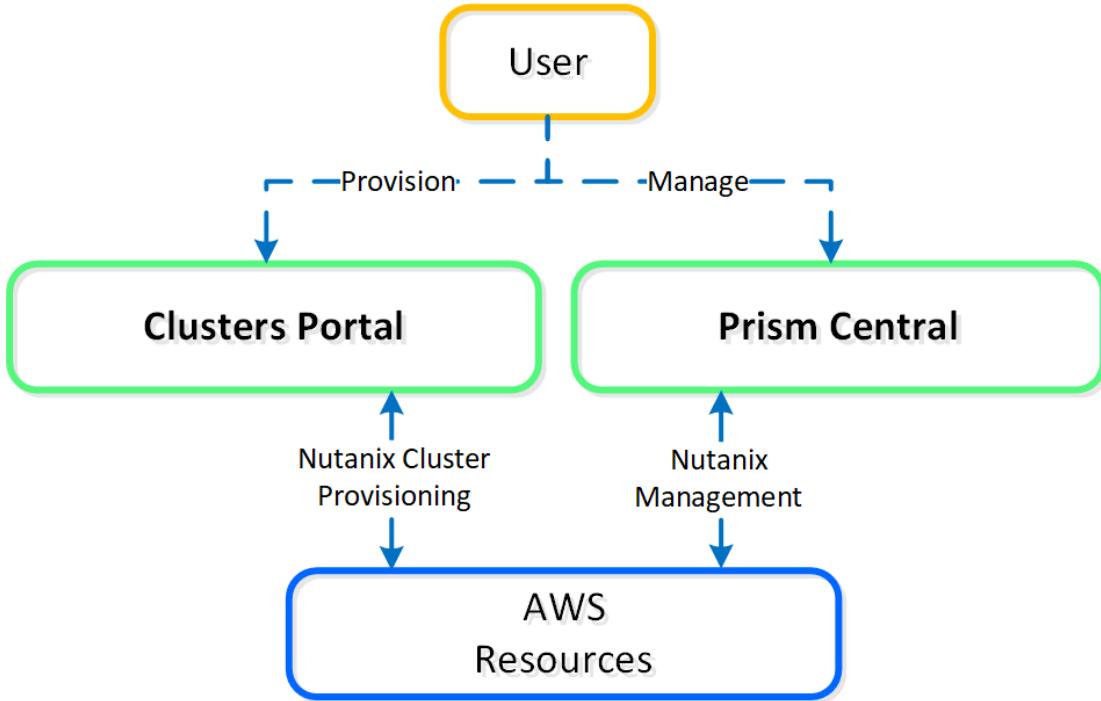


圖. Nutanix Clusters on AWS - Overview

以下內容概述了群集協調器獲取的輸入和一些創建的資源：

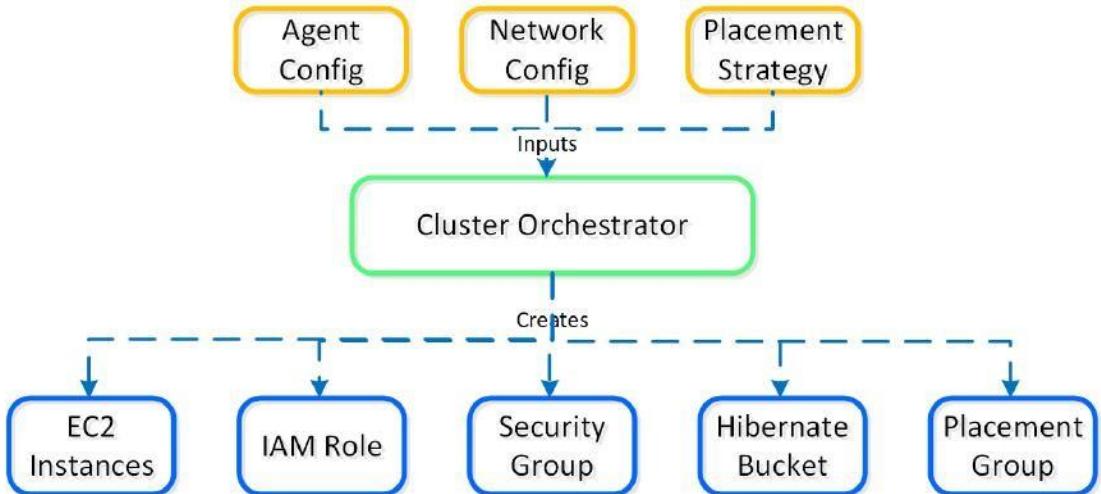


圖. Nutanix Clusters on AWS - Cluster Orchestrator Inputs

下面顯示了 AWS 中節點的高級概述：

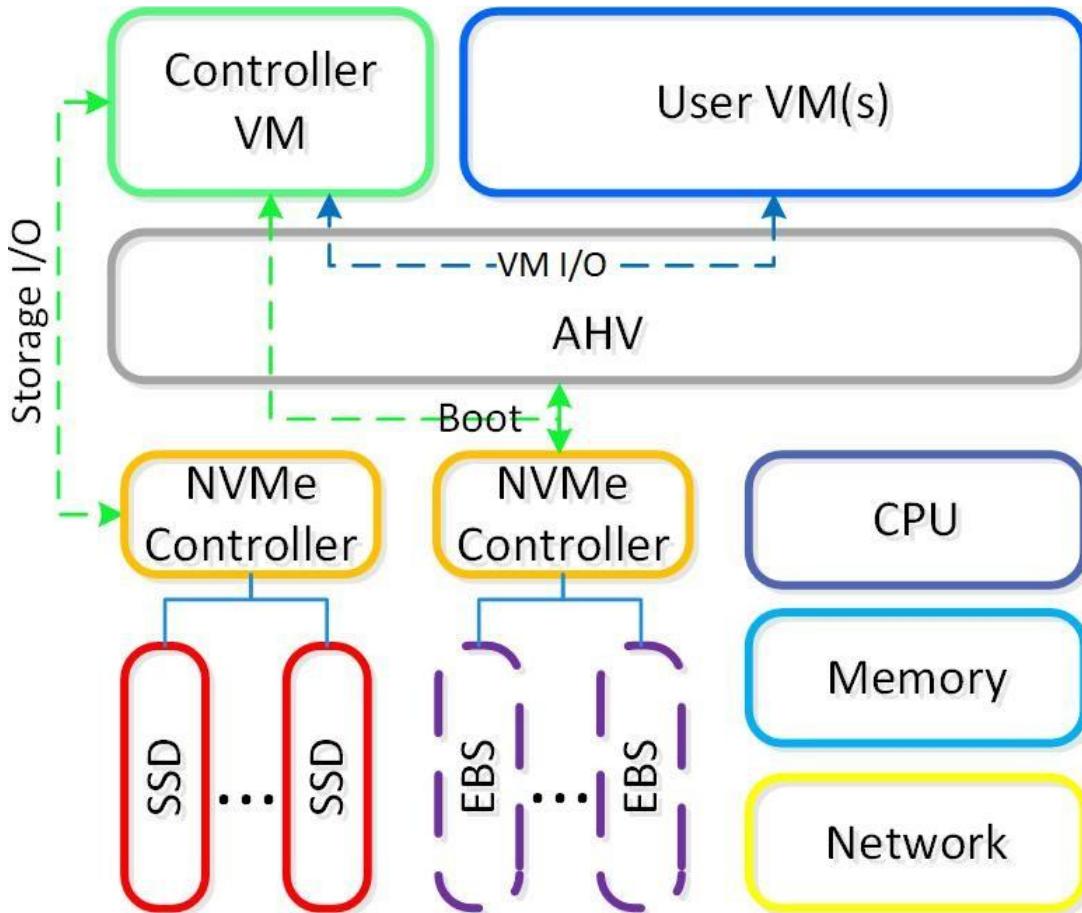


圖. Nutanix Clusters on AWS - Node Architecture

鑑於主機是裸機，我們可以完全控制存儲和網路資源，類似於典型的本地部署。對於 CVM 和 AHV 主機引導，使用 EBS 卷。注意：某些資源（例如 EBS 交互）通過 AWS Nitro 卡運行，該卡在 AHV 主機中顯示為 NVMe 控制器。

8.1.4 擺放策略

預設情況下，AWS 上的 Nutanix 群集使用具有 7 個分區的分區放置策略。主機在與 Nutanix 中的機架相對應的這些分區中劃分為條帶。這樣可以確保您可以出現 1-2 個完整的“機架”故障，並且仍然保持可用性。

下面顯示了分區放置策略和主機條帶化的高層概覽：

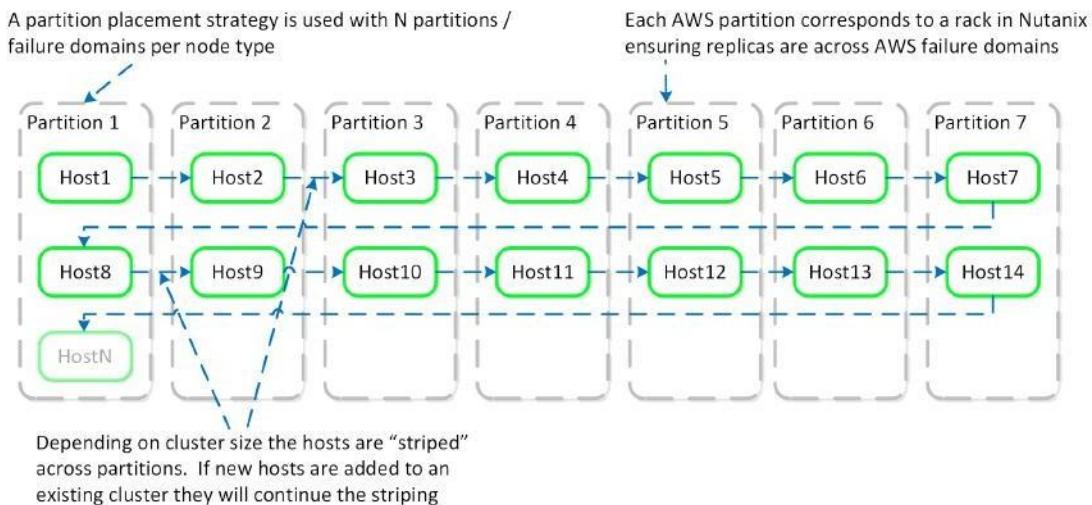


圖. Nutanix Clusters on AWS - Partition Placement

在利用多種節點類型的情況下（例如 i3.metal 和 m5d.metal 等），每種節點類型都有其自己的 7 個分區，節點在這些分區之間進行條帶化。

下面顯示了使用多種實例類型時分區放置策略和主機條帶化的高層概覽：

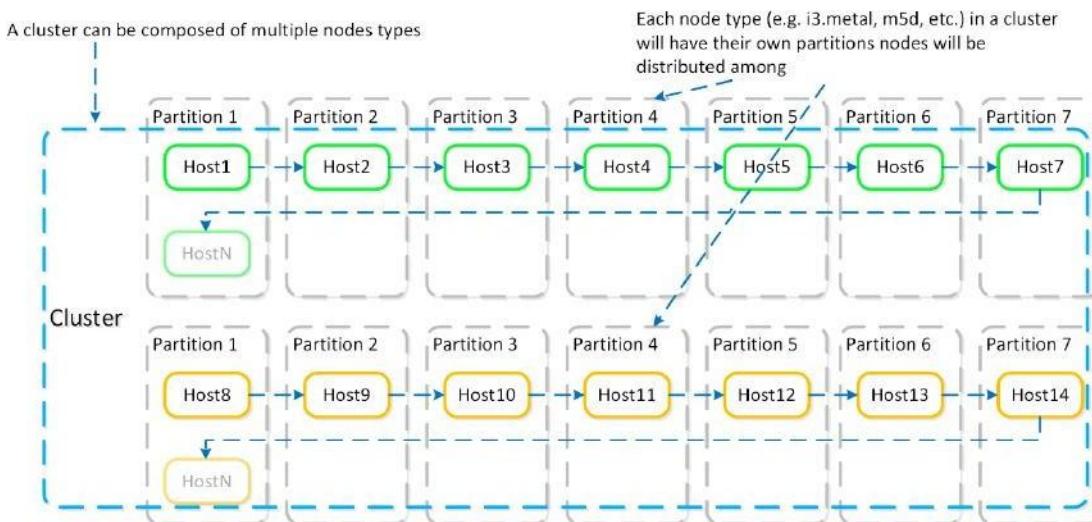


圖. Nutanix Clusters on AWS - Partition Placement (Multi)

8.1.5 存儲

AWS 上的 Nutanix 群集的存儲可以分為兩個核心區域：

1. 核心/活躍
2. 休眠



核心存儲與您在任何 Nutanix 群集中所期望的完全相同，它將“本地”存放裝置傳遞給 CVM，以供 Stargate 使用。

實例存儲

鑑於“本地”存儲由 AWS 實例存儲支援，在斷電/節點故障的情況下，該實例不能完全恢復。必須處理額外的注意事項。

例如，在本地 Nutanix 群集中，如果斷電或節點發生故障，則存儲將保留在本地設備上，並且當節點/電源恢復連線時，存儲將恢復。對於 AWS 實例存儲，情況並非如此。

在大多數情況下，完全可用區應該不會掉電/宕掉，但是對於敏感的工作負載，建議：

- 利用備份解決方案保留到 S3 或任何持久存儲
- 將資料複製到其他 AZ /區域/雲（本地或遠端）中的另一個 Nutanix 群集

AWS 上 Nutanix 群集的一項獨特功能是“休眠”群集的功能，使您可以在關閉 EC2 計算實例的同時持久化資料。這在您不需要計算資源並且不想繼續為它們支付費用，但想要保留資料並能夠在以後還原的情況下很有用。

集群休眠後，資料將從集群備份到 S3。備份資料後，EC2 實例將終止。恢復/還原後，將置備新的 EC2 實例，並將資料從 S3 載入到群集中。

8.1.6 網路連接

網路連接可以分成幾個重要部分：

- 主機/群集網路連接
- 客戶/使用者虛擬機器網路連接
- 廣域網路/L3 網路連接

原生 vs. Overlay

我們決定不運行自己的覆蓋網路，而是決定在 AWS 子網上本地運行，這允許平臺上運行的 VM 與 AWS 服務進行本地通信，沒有性能下降。

AWS 上的 Nutanix 群集已置備到 AWS VPC 中，以下內容顯示了 AWS VPC 的高層概覽：

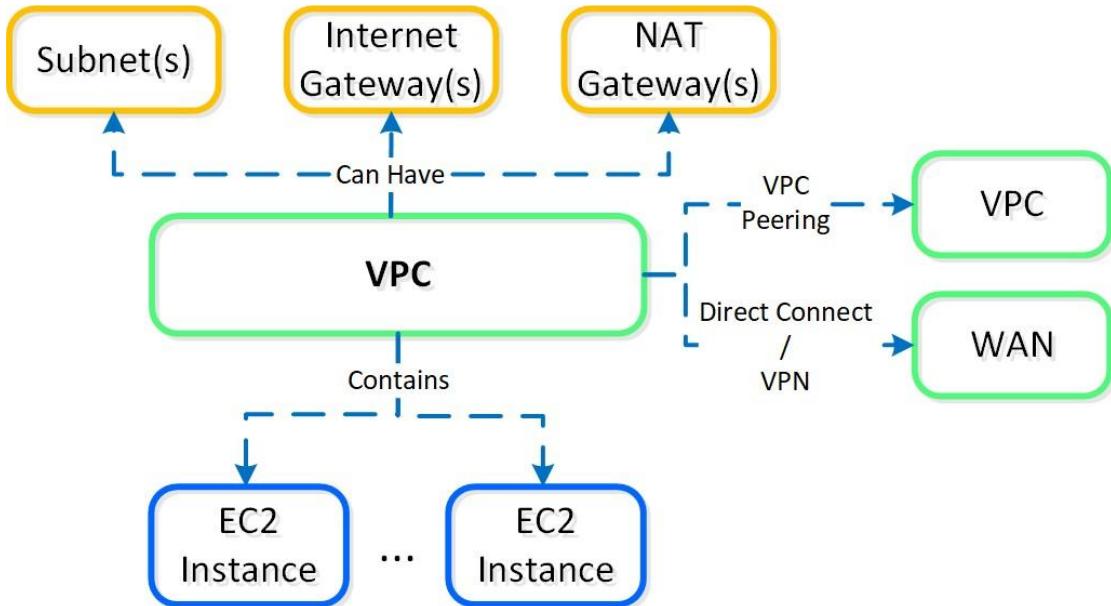


圖. Nutanix Clusters on AWS - AWS VPC

新的 VPC vs. 默認的 VPC

AWS 將創建默認的 VPC / Subnet / Etc。每個區域都有 172.31.0.0/16 ip 方案。

建議使用適合您公司 IP 方案的關聯子網，NAT / Internet 閘道等創建新的 VPC。如果您計畫在 VPC 之間（VPC 對等）或到現有 WAN 擴展網路，則這一點很重要。我將其視為 WAN 上的任何網站。

8.1.7 主機網路連接

在 AWS 裸機上運行的主機是傳統的 AHV 主機，因此可以利用相同的基於 OVS 的網路堆疊。

下面顯示了 AWS AHV 主機的 OVS 堆疊的高層概覽：

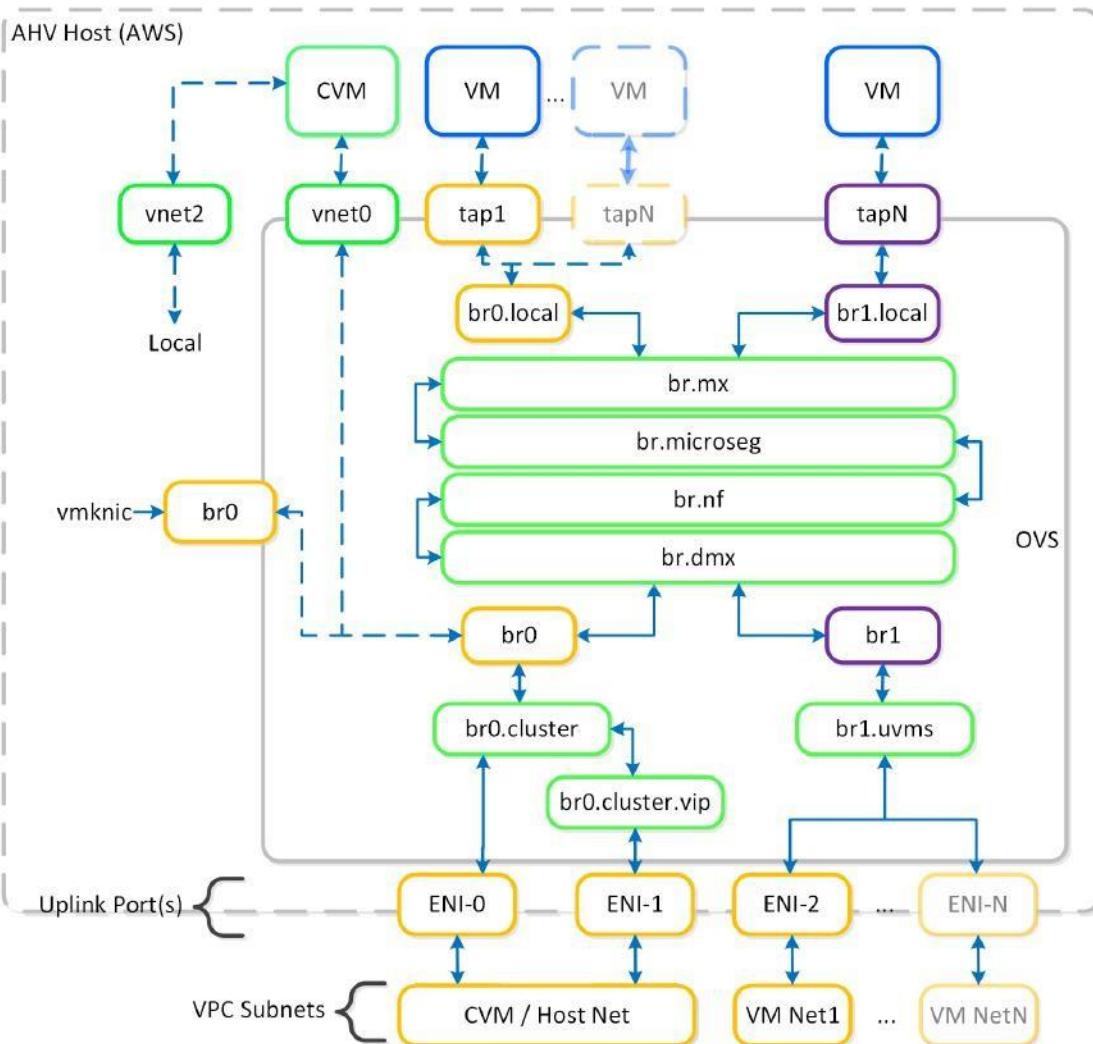


圖. Nutanix Clusters on AWS - OVS Architecture

除了添加了 L3 上行鏈路橋接器之外，OVS 堆疊與任何 AHV 主機都基本相同。對於 UVM（來賓虛擬機器）網路，使用 VPC 子網。可以在群集創建過程中或通

過以下步驟創建 UVM 網路：

在 AWS VPC 儀錶板中，按一下“子網”，然後按一下“創建子網”，然後輸入網路詳細資訊：



Name tag i

VPC* ▼ i

VPC CIDRs
172.20.0.0/16
2a05:d01c:b55:d800::/56

Availability Zone ▼ i

IPv4 CIDR block* i

IPv6 CIDR block ▼ i

圖. Nutanix Clusters on AWS - OVS Architecture

注意：CIDR 塊應該是 VPC CIDR 範圍的子集。

子網將從 VPC 繼承路由表：

Subnet: subnet-

Description Flow Logs **Route Table** Network ACL Tags Sharing

Edit route table association

Route Table: rtb-

1 to 4 of 4 < >

Destination	Target
172.10.0.0/16	pcx- -----
172.20.0.0/16	local
0.0.0.0/0	igw- -----
2a05:d01c:b55:d800::/56	local

圖.AWS 上的 Nutanix 群集-路由表

在這種情況下，您可以看到對等 VPC 中的任何流量都將通過 VPC 對等連結，而任何外部流量都將通過 Internet 閘道。



為了告訴 AOS 使用此網路，我們將標籤添加到子網中。選擇子網，然後按一下“添加/編輯標籤”：

The screenshot shows the AWS VPC Dashboard. On the left, there's a sidebar with 'Virtual Private Cloud' and 'Your VPCs' sections, and 'Subnets' is currently selected. In the main area, a specific subnet is listed with fields for 'Name' (Sample-UVM) and 'Sample-UVM'. To the right, a context menu is open under the 'Actions' heading, with 'Add/Edit Tags' highlighted.

圖.AWS 上的 Nutanix 群集-添加/編輯標籤

添加金鑰為“ntnx : vlan”和該網路的 VLAN ID 的新標籤：

The screenshot shows the 'Add/Edit Tags' dialog box. It has a header 'Add/Edit Tags' and a sub-instruction 'Apply tags to your resources to help organize and identify them.' Below this, it explains what a tag is: 'A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.' It includes a link to 'Learn more' about tagging. The main area is a table with columns 'Key' and 'Value'. There are two entries: one for 'Name' with 'Sample-UVM' as the value, and another for 'ntnx:vlan' with '35' as the value. At the bottom are 'Create Tag', 'Cancel', and 'Save' buttons.

圖.AWS 上的 Nutanix 群集-添加/編輯標籤

一旦子網在 AWS 端看起來不錯，請使用以下命令“`subnet-manager-cli list-networks`”連接到 CVM 並發現新網路。這將查詢 VPC 並列出子網。



```
[nutanix@NTNX-172-20-10-8-A-CVM:172.20.10.8:~$ subnet-manager-cli list-networks
AWS subnet id          CIDR block      Acropolis network id      Can be added
subnet-                172.20.10.0/24
subnet-                172.31.16.0/20
subnet-                172.20.20.0/24  10fd1b6a-2d57-49e5-a4b6-147305413f11  true
subnet-                172.31.0.0/20
subnet-                172.31.50.0/24
subnet-                172.20.35.0/24
subnet-                172.31.49.0/24
subnet-                172.31.48.0/24
subnet-                172.31.32.0/20
nutanix@NTNX-172-20-10-8-A-CVM:172.20.10.8:~$ ]
```

圖.AWS 上的 Nutanix 群集-路由表

獲取 AWS 子網 ID 後，使用以下命令“`allssh aws-subnet-manager add-network <SUBNET ID>`”添加到 Acropolis。

```
[nutanix@NTNX-172-20-10-8-A-CVM:172.20.10.8:~$ subnet-manager-cli add-network subnet-
2019/09/12 03:45:48 adding 172.20.35.0/24 to acropolis
2019/09/12 03:45:51 subnet 172.20.35.0/24 added to acropolis
nutanix@NTNX-172-20-10-8-A-CVM:172.20.10.8:~$ ]
```

圖.AWS 上的 Nutanix 群集-添加網路

完成後，您將在 Prism 中看到該網路可用。

8.1.8 WAN / L3 網路

在大多數情況下，部署將不僅僅在 AWS 中，還需要與外部世界（其他 VPC，Internet 或 WAN）進行通信。

要連接 VPC（位於相同或不同區域），可以使用 VPC 對等連接，該對等連接允許您在 VPC 之間建立隧道。注意：您需要確保遵循 WAN IP 方案的最佳做法，並且 VPC /子網之間沒有 CIDR 範圍重疊。

下圖顯示了 eu-west-1 和 eu-west-2 區域中的 VPC 之間的 VPC 對等連接：

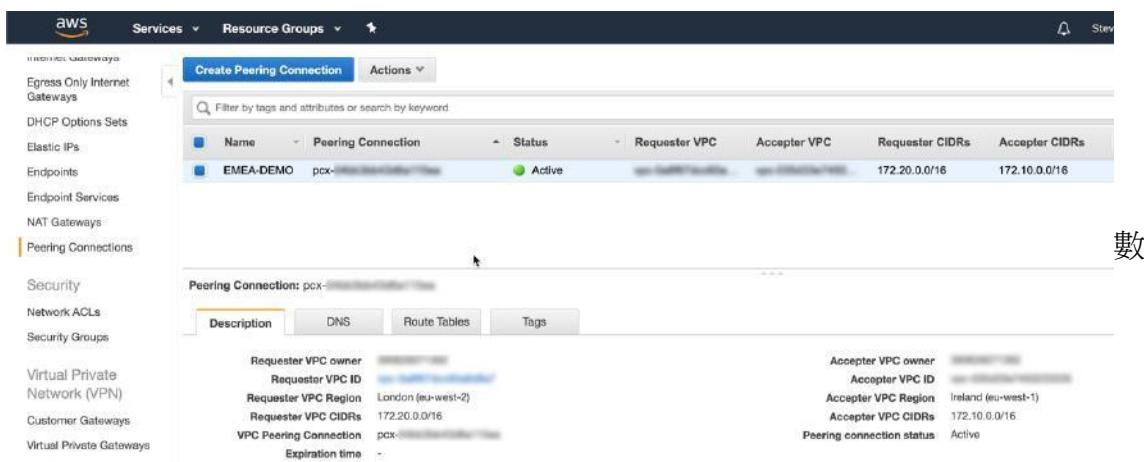


圖.AWS 上的 Nutanix 群集-VPC 對等



然後，每個 VPC 的路由表都將通過對等連接路由去往另一個 VPC 的流量（如果通信需要是雙向的，則該流量表將在兩端都存在）：

The screenshot shows the AWS Route Table configuration for a specific subnet. The top navigation bar includes tabs for Description, Flow Logs, Route Table (which is selected and highlighted in orange), Network ACL, Tags, and Sharing. Below the tabs, there is a blue button labeled "Edit route table association". The main content area is titled "Route Table: rtb-XXXXXX" and displays a table of routes. The table has two columns: "Destination" and "Target". The routes listed are:

Destination	Target
172.10.0.0/16	pcx-XXXXXX
172.20.0.0/16	local
0.0.0.0/0	igw-XXXXXX
2a05:d01c:b55:d800::/56	local

圖.AWS 上的 Nutanix 群集-路由表

為了將網路擴展到本地/ WAN，可以利用 VPN 閘道（tunnel）或 AWS Direct Connect。

8.1.9 安全

鑑於這些資源在我們完全控制的安全性之外的雲中運行，因此資料加密和合規性是非常關鍵的考慮因素。

這些建議的特徵如下：

- 啟用資料加密
- 僅使用專用子網（不分配公共 IP）
- 鎖定安全性群組和允許的埠/ IP CIDR 塊
- 要獲得更精細的安全性，請利用 Flow

8.1.10 用法和配置

以下各節介紹如何在 AWS 上配置和利用 Nutanix 群集。

高層過程可以分為以下高層步驟：

- 創建 AWS 帳戶
- 配置 AWS 網路資源（如有必要）
- 通過 Nutanix 群集門戶配置群集
- 預配完成後利用集群資源



還有更多！

9 第九部分：存儲服務

9.1 塊服務

Acropolis Block Services (ABS) 的功能就是通過 iSCSI 提供後端的 DSF 存儲給外部的使用者（guest OS，物理主機，容器等）。

這將允許任何作業系統能夠直接訪問 DSF 利用存儲的功能。在這種部署場景下，作業系統繞過任何 Hypervisor 直接和 Nutanix 對話。

核心的 Acropolis Block Services 使用案例：

- 共用磁片
 - Oracle RAC，Microsoft Failover Clustering 等
- 首選的存儲磁片
 - 執行環境生命週期短但資料關鍵
 - 容器 s，OpenStack 等
- 用戶端發起的 iSCSI
 - 當裸盤使用
 - Exchange on vSphere (支持 Microsoft)

認證的作業系統

根據 iSCSI 相容性要求，以下作業系統通過了認證：

- Microsoft Windows Server 2008 R2/2012 R2
- Redhat Enterprise Linux 6.0+

塊服務的概念

以下組件構成了 Acropolis Block Services

- Data Services IP : iSCSI 登錄請求時使用的集群 IP 位址（4.7 章節介紹）
- Volume Group : iSCSI 目標和磁片組，在集中管理，快照和策略申請時使用
 - Disk(s) : 添加在 Volume Group 中的磁片設備（就像 iSCSI 目標中的 LUNs）
 - Attachment : 允許一個特定的發起者針對 Volume Group 的 IQN 准入
 - Secret(s) :

NOTE: 在後端，一個 VG 的 disk 就是一個在 DSF 上的 vDisk.

前提條件



在配置之前，需要先配置作為集中發現/登錄門戶的 Data Services IP，可以通過‘Cluster Details’頁面(Gear Icon -> Cluster Details)來設置。

CLUSTER NAME
TMBEAST

CLUSTER VIRTUAL IP ADDRESS
10.3.140.100

EXTERNAL DATA SERVICES IP ADDRESS
10.3.140.99

Cancel Save

圖.塊服務-資料服務 IP

也可以通過 NCLI/API:

```
ncli cluster edit-params external-data- services-ip-address=<DATA SERVICES IP ADDRESS>
```

創建目標

在使用塊服務之前，我們需要先創建一個‘Volume Group’作為 iSCSI 的目標
在‘Storage’頁點擊右手邊的‘+ Volume Group’

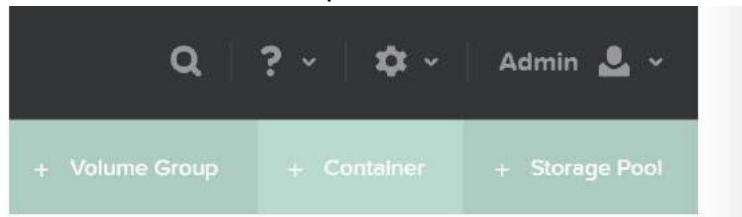


圖. 塊服務-添加卷組

在展開的功能表中需要指定 VG 的詳細資訊：

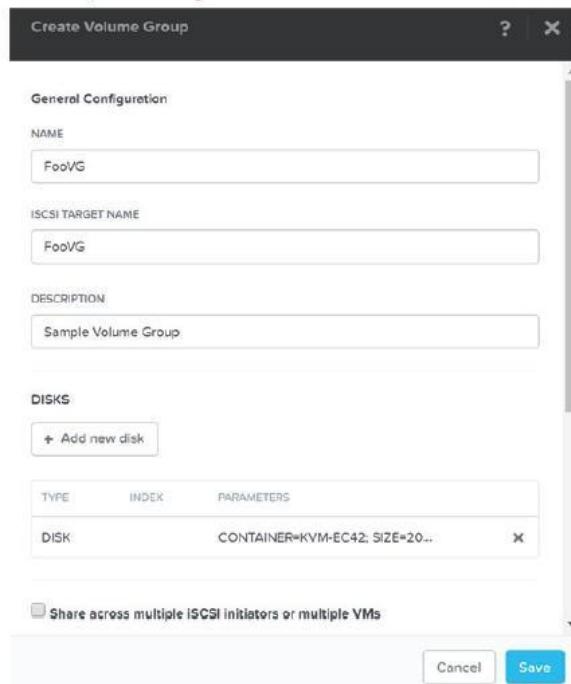


圖.塊服務-添加 VG 細節

然後點擊‘+ Add new disk’添加任何磁片到目標中。在隨後出現的功能表中允許我們選擇目標容器和磁片大小。

圖.塊服務-添加磁片

點擊‘Add’。如果你需要添加多個磁片，可以重複這個步驟。



當我們制定詳細資訊並添加完磁片，我們可以將 Volume Group 附加到虛擬機器或者 IQN 的發起者，這將允許虛擬機器訪問 iSCSI 目標（來自不明發起者的請求會被拒絕）

The screenshot shows the 'INITIATORS' section with an 'IQN' input field containing 'iqn.1991-05.com.microsoft:desktop-p0q1a3'. Below it, the 'VMs' section has a '+ Attach to a VM' button. At the bottom right are 'Cancel' and 'Save' buttons.

圖. 塊服務 – 發起者 IQN/VM

點擊‘Save’，Volume Group 配置完成。

以上配置也可以通過 ACLI／API 來完成：

#創建 VG

```
vg.create <VG Name>
```

#添加磁片到 VG

```
Vg.disk_create <VG Name> container=<CTR Name> create_size=<Disk size, e.g. 500G>
```

#附加發起者 IQN 到 VG

```
Vg.attach_external <VG Name> <Initiator IQN>
```

路徑 HA

前面提到，Data Services IP 讓我們在不需要知道私有的 CVM IP 位址的情況下也能進行發現操作。

Data Services IP 被分配到當前的 iSCSI Master，一旦出現故障，一個新的 iSCSI Master 會被選舉出來且自動分配這個 Data Services IP。這種機制確保了 discovery portal 始終可用。

配置了 Data Services IP 的 iSCSI 發起者會作為 iSCSI 目標入口。當有一個登錄請求時，平臺會執行一個重定向到健康 Stargate 的 iSCSI 登錄。

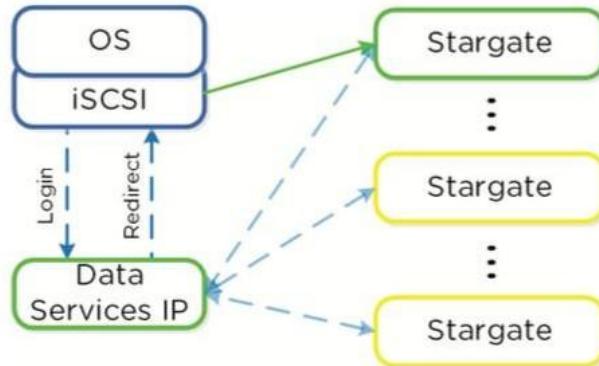


圖.Block Services - 重定向登錄

當出現活動的關聯 Stargate 失效時，發起者會嘗試讓 iSCSI 登錄到 Data Services IP。這時，Data Services IP 將登錄重定向到另一個健康的 Stargate。

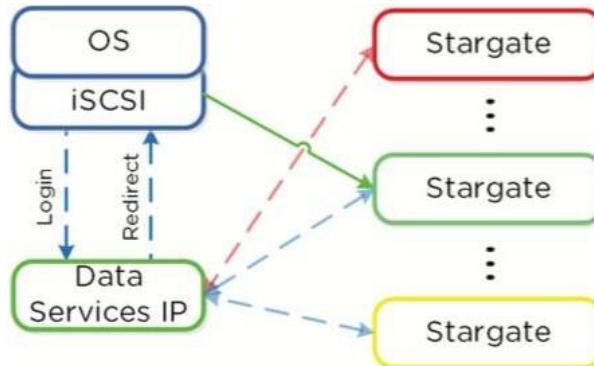


圖.Block Services - 故障接管

當出現問題的關聯 Stargate 恢復正常且可用，當前活動的 Stargate 會靜默 I/O 並結束活動的 iSCSI 會話。當發起者重新嘗試進行 iSCSI 登錄時，Data Services IP 將登錄重定向到回復正常的關聯 Stargate。

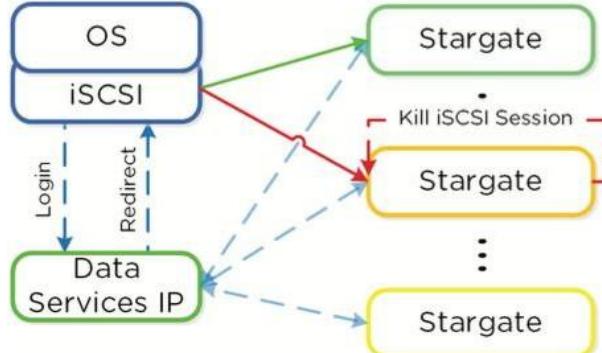
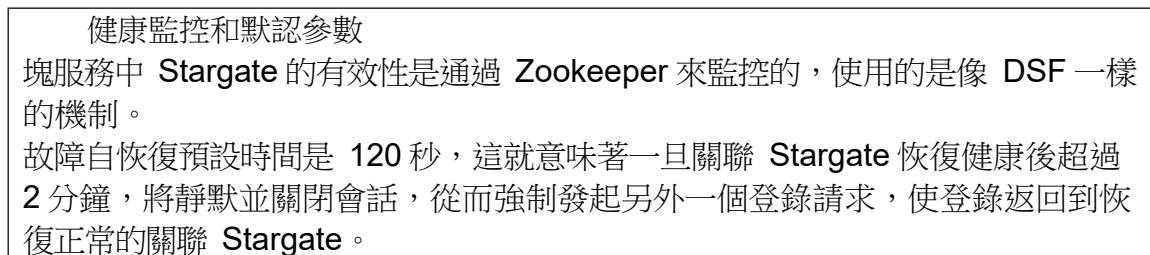


圖.塊服務 - 自動恢復



通過這種機制，用戶端多路徑（MPIO）不再需要 **path HA**。現在我們不需要在連接到發起者的同時去檢查“**Enable multi-path**”選項（這個選項是啟用 MPIO 的）

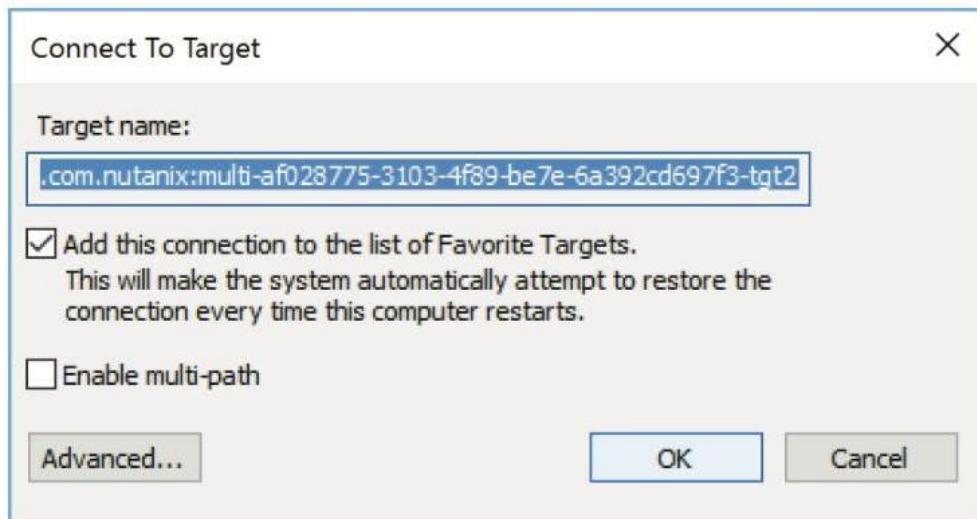


圖.塊服務 - 無需 MPIO 多路徑

iSCSI 協定為每個目標在發起者和目標之間設置一個單獨的 iSCSI 會話（TCP 連結）。這就是說，**Stargate** 和目標之間是一對一的關係。
如 4.7 章節所描述，默認有 32 個虛擬的目標在附加發起者和將每個磁片添加到 **volume group (VG)** 時自動創建完成。當創建多個 VG，並且每個 VG 只包含一個磁片時，每個磁片會提供一個 iSCSI 目標。
當我們通過 **ACLI/ API** 查看 VG 詳細資訊時，你會看到為每個 **attachment** 創建出來的 32 個虛擬目標。

```
attachment_list {
    external_initiator_name: "iqn.1991-05.com.microsoft:desktop-foo"
    target_params {
        num_virtual_targets: 32
    }
}
```

在這裡，我們已經創建出來一個添加了 3 個磁片設備的 VG，當在用戶端發起搜索時，我們會看到每個磁片設備都有一個自己的目標。（使用‘-tgt[int]’尾碼）

Discovered targets

Name	Status
iqn.2010-06.com.nutanix:multi-af028775-3103-4f89-be7e-6a392cd697f3-tgt0	Conne
iqn.2010-06.com.nutanix:multi-af028775-3103-4f89-be7e-6a392cd697f3-tgt1	Conne
iqn.2010-06.com.nutanix:multi-af028775-3103-4f89-be7e-6a392cd697f3-tgt2	Conne

Refresh

To connect using advanced options, select a target and then click Connect.

To completely disconnect a target, select the target and then click Disconnect.

For target properties, including configuration of sessions, select the target and click Properties... .

For configuration of devices associated with a target, select the target and then click Devices... .

Connect

Disconnect

Properties...

Devices...

圖.塊服務 - 虛擬目標

這樣每個磁片設備都有一個自己的 iSCSI 會話，並且這些會話能夠在多個 Stargate 之間調度，增加可擴展性和提高了性能。

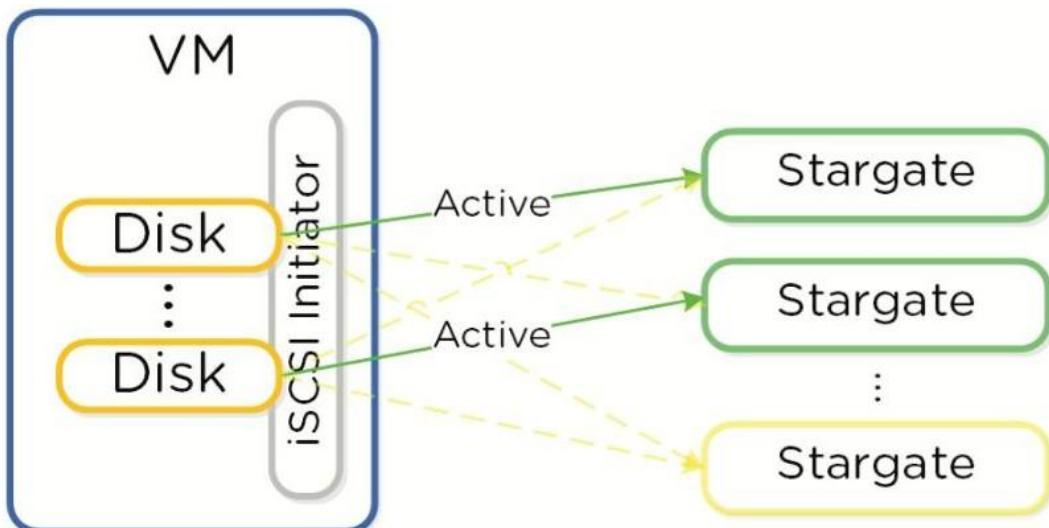


圖.塊服務 - 多路徑

當為每個目標都建立了 iSCSI 會話(iSCSI 登錄)，負載均衡就實現了。

活動路徑

使用以下命令查看駐留虛擬目標的活動 Stargate(s) :

```
# Windows  
Get-NetTCPConnection -State Established -RemotePort 3205  
  
# Linux  
iscsiadm -m session -P 1
```

如 4.7 章節說說，當分配目標要橫跨集群節點時，如果需要，我們會用到 hash 演算法。當我們要在健康節點中設置一個優先節點時到也會用到 hash 演算法。

SCSI UNMAP (TRIM)

當我們需要從已刪除的資料塊中回收存儲空間時，Acropolis Block Services 支援符合 SCSI T10 規格的 SCSI UNMAP (TRIM) 命令列。

9.2 檔服務

檔服務特性允許使用者將 Nutanix 平臺當作一個高可用的檔案伺服器。使用者可以在一個單一命名空間中存儲主目錄和檔。

支援的配置

檔服務支援如下配置（清單不完整，請參考手冊獲取完全的支持列表）：虛擬化層

- AHV
- ESXi

檔協議：

- CIFS2.1

相容特性：

- Async-DR

這個特性由以下一系列高級別的概念構成：

- 檔案伺服器
 - 高級別命名空間。每個檔案伺服器擁有一組檔服務虛擬機器。
- 共用
 - 開放給用戶的共用。一個檔案伺服器可以擁有多個共用（例如部門級共用等等）
- 資料夾

存儲檔的資料夾。資料夾在檔服務虛擬機器間共用下

圖是這些高級別概念之間的映射關係：

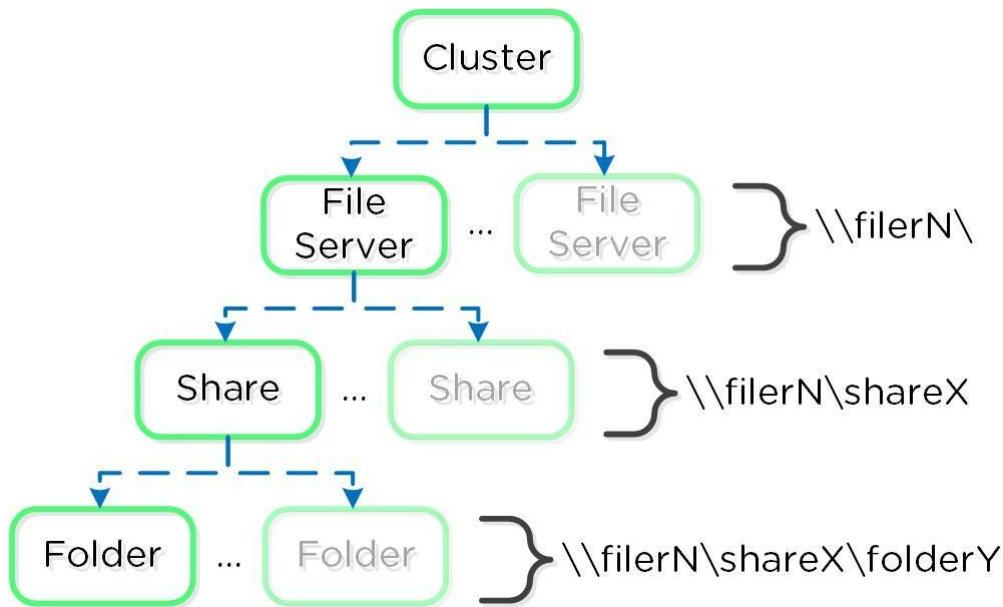


圖. 檔服務關係映射

檔服務特性遵從與 Nutanix 平臺相同的分散式方法論，確保可用性和擴展性。檔案伺服器部署需要最少三個檔服務虛擬機器
下圖是這些組件的細節：

The file services feature is composed of multiple File Services VMs (FSVM) for distribution and scale

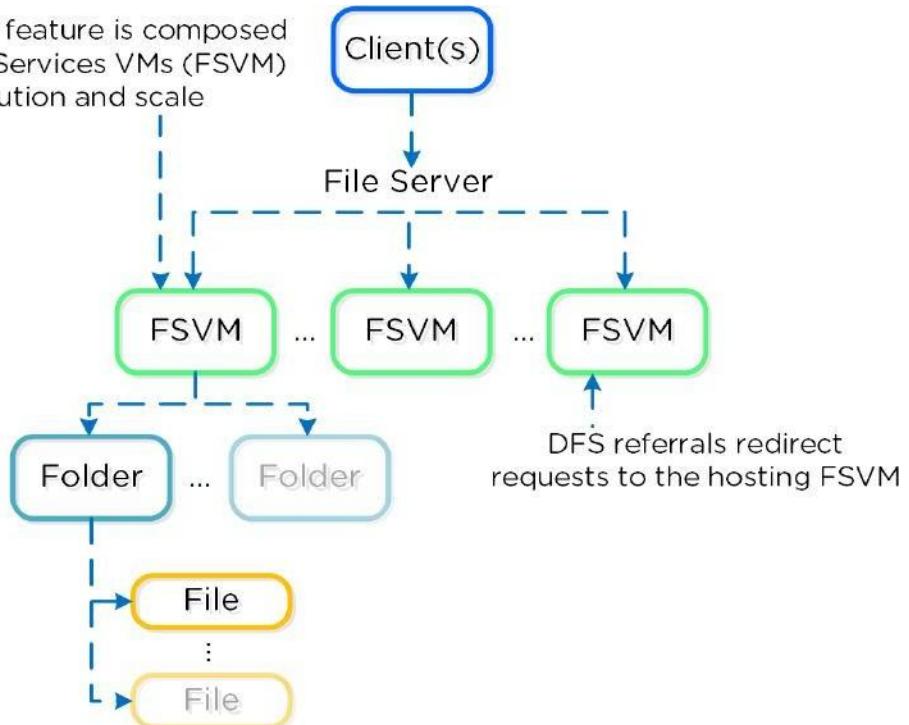


圖. 檔服務的細節

支援的協定

4.6 版本中，SMB（最高到 2.1 版本）是唯一受支援的用戶端與檔服務通訊的協定。

檔服務虛擬機器作為代理虛擬機器運行在平臺上，並且是在配置過程中被透明地部署的。

下圖是 Acropolis 平臺上檔服務虛擬機器的細節：

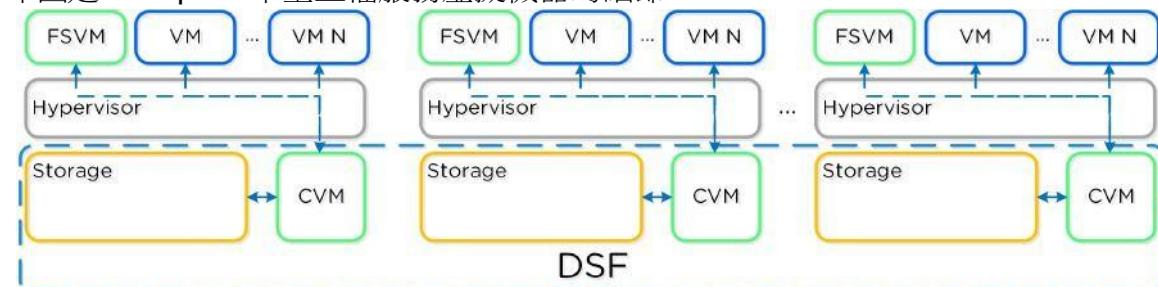


圖. 檔服務虛擬機器部署架構

認證與授權

檔服務特性與微軟活動目錄（AD）和 DNS 完整集成。這樣就可以利用 AD 中所有安全的和已有的認證和授權功能。所有的共用許可權、用戶和組管理也可以在傳統的負責 Windows 檔管理的 MMC 中完成。在安裝過程中，會創建下列 AD/DNS 物件：

- 檔案伺服器的 AD 電腦帳戶
- 檔案伺服器和每個檔服務虛擬機器的 AD Service Principal Name (SPN)
- 指向所有檔服務虛擬機器的檔案伺服器 DNS 條目
- 每個檔服務虛擬機器的 DNS 條目

用於檔案伺服器創建的 AD 特權

因為需要創建 AD 和 DNS 物件，所以必須使用域管理員或同等特權的使用者帳戶來部署檔服務特性。

高可用 (HA)

每個檔服務虛擬機器借助 Acropolis Volumes API 通過 in-guess iSCSI 來訪問自己的資料存儲。當檔服務虛擬機器宕機時，任意檔服務虛擬機器就可以連接任意 iSCSI 目標。

下圖是檔服務虛擬機器存儲的上層概覽：

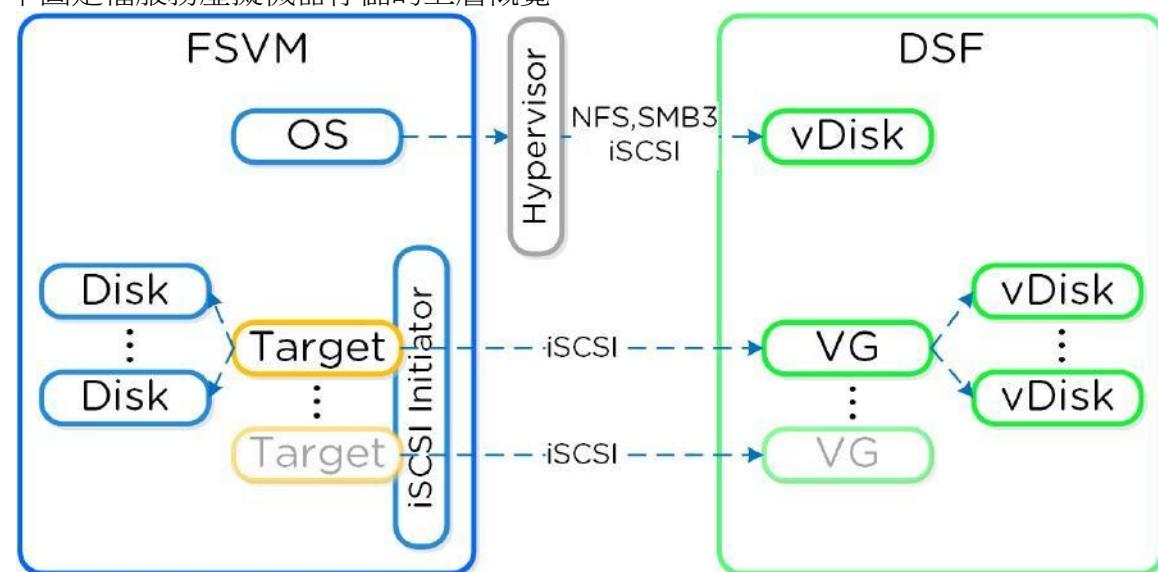


圖. 檔服務虛擬機器存儲

檔服務虛擬機器借助 DM-MPIO 提供路徑高可用，DM-MPIO 預設將活動路徑設置在本地 CVM：

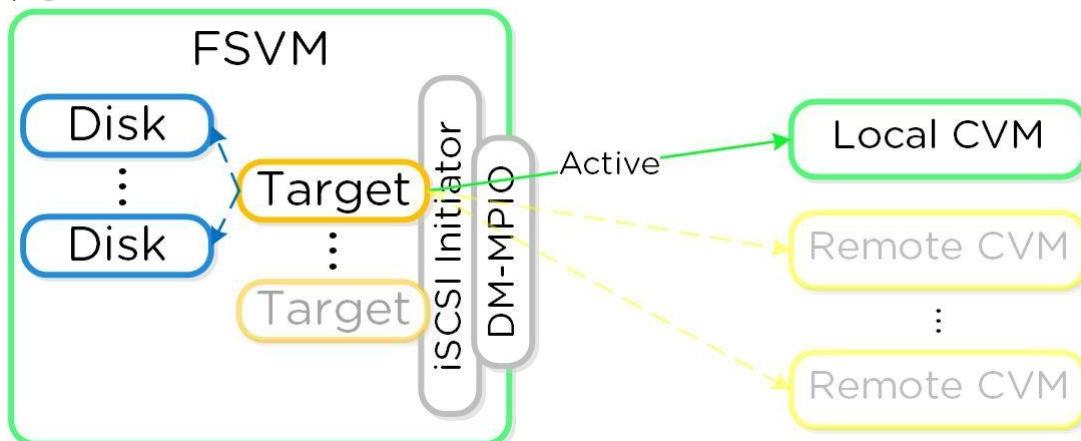


圖. 檔服務虛擬機器 MPIO

當本地 CVM 不可用（例如活動路徑中斷）時，DM-MPIO 會啟動一條到遠端 CVM 的容錯移轉路徑，這條路徑會繼續接管處理 IO。

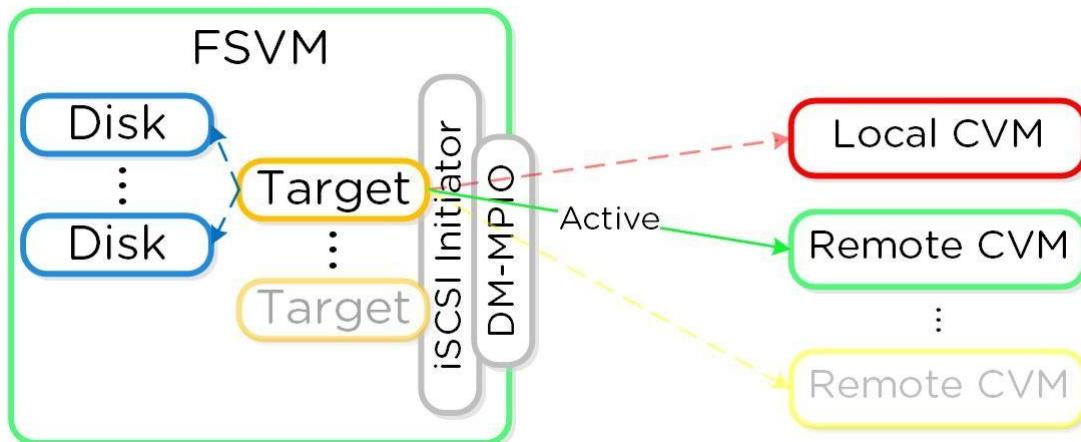


圖. 檔服務 MPIO 容錯移轉

當本地 CVM 恢復正常並且健康時，它會被標記為活動路徑為本地 IO 提供服務。正常工作環境中，每個檔服務虛擬機器都會與自己的作為資料存儲的 VG 通訊，並保持到其它 VG 的備用連接。作為 DFS 受訪進程的一部分，每個檔服務虛擬機器都有一個供用戶端與自身通訊的 IP。但是用戶端不需要知道每個獨立的檔服務虛擬機器的 IP，因為 DFS 受訪進程會連接用戶端到承載它們資料夾的正確 IP。

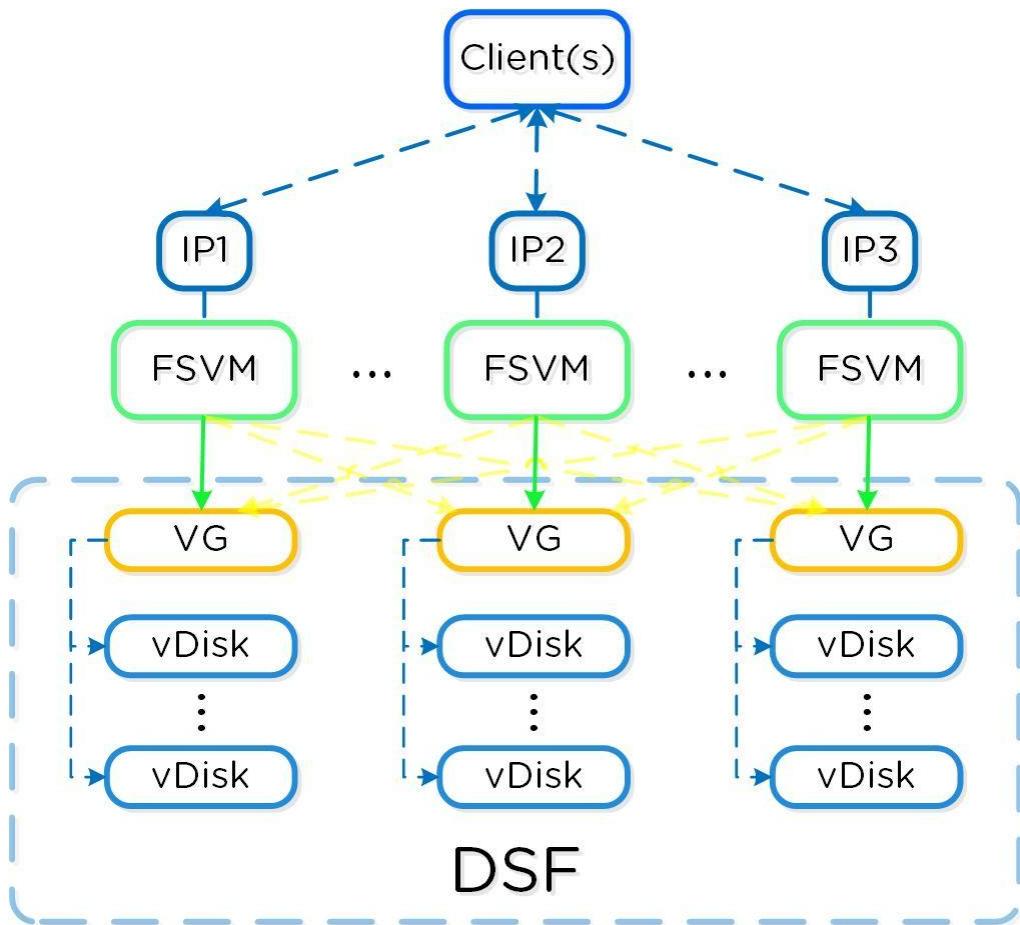


圖. 檔服務虛擬機器的工作模式

當檔服務虛擬機器“故障”（例如人工維護、斷電等）時，故障檔服務虛擬機器的 VG 和 IP 都會被另一個檔服務虛擬機器接管，確保用戶端的可用性。

下圖是故障檔服務虛擬機器的 IP 和 VG 轉移過程：

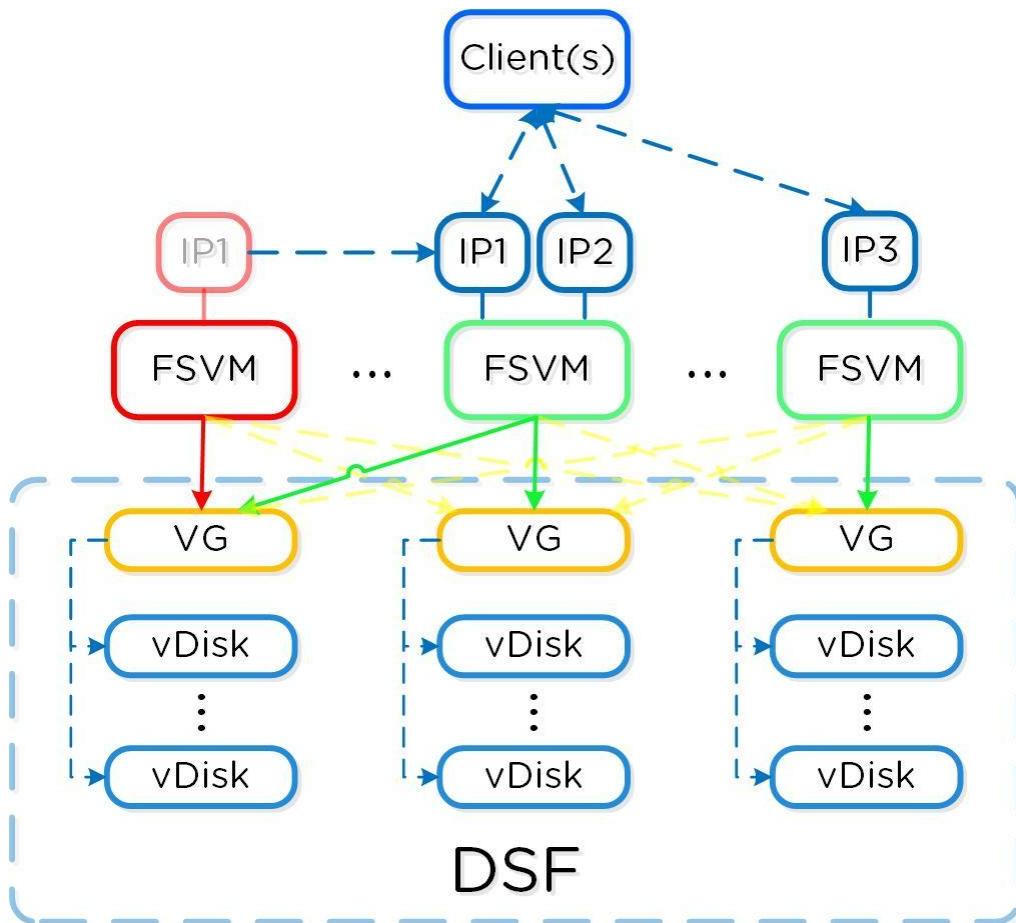


圖. 檔服務虛擬機器故障場景

當故障檔服務虛擬機器恢復正常並且穩定時，它會重新接管自己的 IP 和 VG 繼續為用戶端 IO 提供服務。

9.3 物件存儲服務

Nutanix Objects 功能通過一個符合 S3 的 API 提供了高度可擴展的持久物件服務（有關 S3 的更多資訊：[LINK](#)）。由於 Nutanix Objects 已部署在 Nutanix 平臺之上，因此它可以利用 AOS 功能，如重複資料刪除，壓縮，複製等。Objects 是在 AOS 5.11 中引入的。

支援的配置

該解決方案適用於以下配置（清單可能不完整，請參閱文檔以獲取完全支持的列表）：

核心用例：



- 備份
- 大資料/分析

管理介面：

- Prism Central (PC)

Hypervisor(s)：

- N/A-運行在 Nutanix MSP 的。（取決於受 MSP 支援的 Hypervisors）

升級：

- LCM

相容功能：

- TBI

Object 協議：

- S3 (版本 4)

Nutanix 微服務平臺 (MSP)

Nutanix Object 利用 Nutanix 微服務平臺 (MSP)，並且是這樣做的首批核心服務之一。

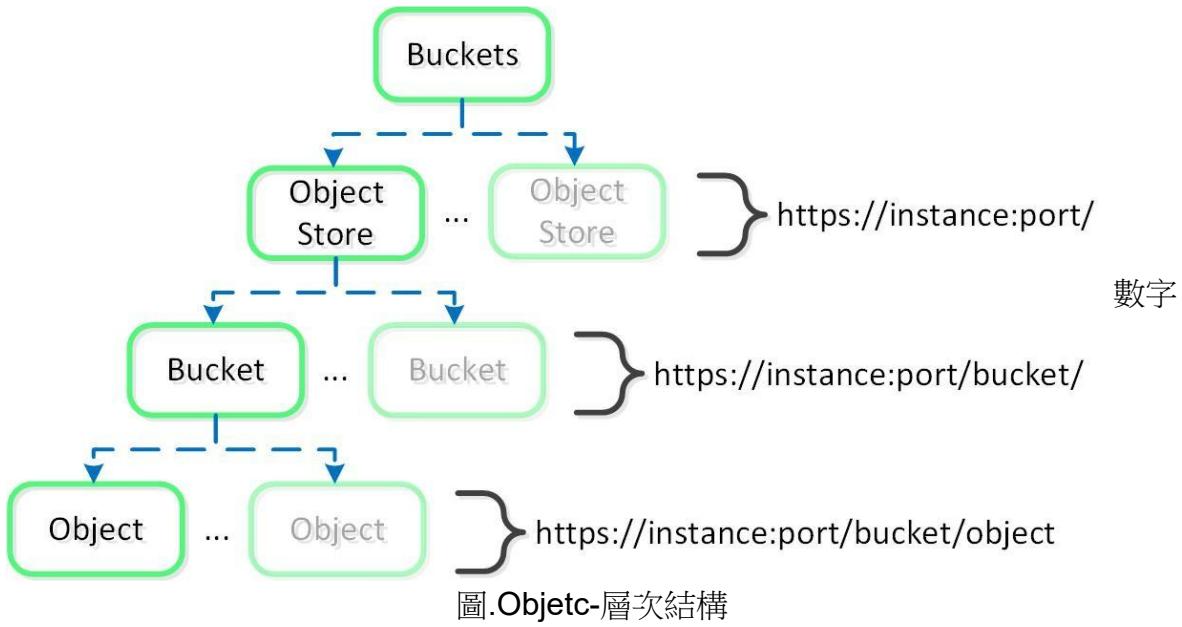
Nutanix MSP 提供了一個通用框架和服務來部署 Object 元件的關聯容器和平臺服務，例如身份和訪問管理 (IAM) 和負載平衡 (LB)。

關鍵條款

在本節中將使用以下關鍵術語，並在以下術語中進行定義：

- Bucket
 - 向使用者公開並包含 Object 的組織單位（與檔案伺服器上的檔共用）。一個部署可以而且通常會有多個 Bucket（例如部門，分區等）。
- Object
 - 通過 API (GET/PUT) 介面的存儲和項的實際單元(blob)
- S3
 - 該術語用於描述引入的原始物件服務 Amazon Web Services (AWS)。現在在語法上用於物件服務。S3 還用於定義在整個項目中都得到高度利用的 Object API。

該圖顯示了概念結構的高級映射：



Object 構造

此功能由一些高級構造組成：

- **Load Balancer**
 - 負載平衡器是 Nutanix MSP 的一部分，並充當服務和資料請求的代理。這樣可以確保 Object 容器之間的服務和負載平衡具有高可用性。
- **Service Manager**
 - 服務管理器充當所有 UI 請求的端點，並管理 Object 存儲實例。它還負責從實例收集統計資訊。
- **Metadata Server**
 - 中繼資料伺服器負責包含 Nutanix Object 部署周圍的所有元資訊（例如 buckets, Object 等）。利用 ChakrDB，這是 Nutanix 開發的基於 RocksDB 的鍵值存儲。ChakrDB 使用 Nutanix ABS 進行存儲。
- **Object Controller**
 - Object Controller 負責管理物件資料並與中繼資料伺服器協調中繼資料更新。它通過存儲代理 API 與 Stargate 連接。
- **Region Manager**
 - Region Manager 負責管理 Acropolis DSF 上的所有物件存儲資訊（例如，區域）。
- **Region**
 - Region 提供 Object 與 Nutanix vDisk 上相應位置之間的高級映射。類似於虛擬磁片 ID，偏移量和長度。
- **Atlas Service**
 - Atlas Service 負責 Object 生命週期策略的實施和執行垃圾收集。

該圖顯示了 Object 服務體系結構的詳細視圖：

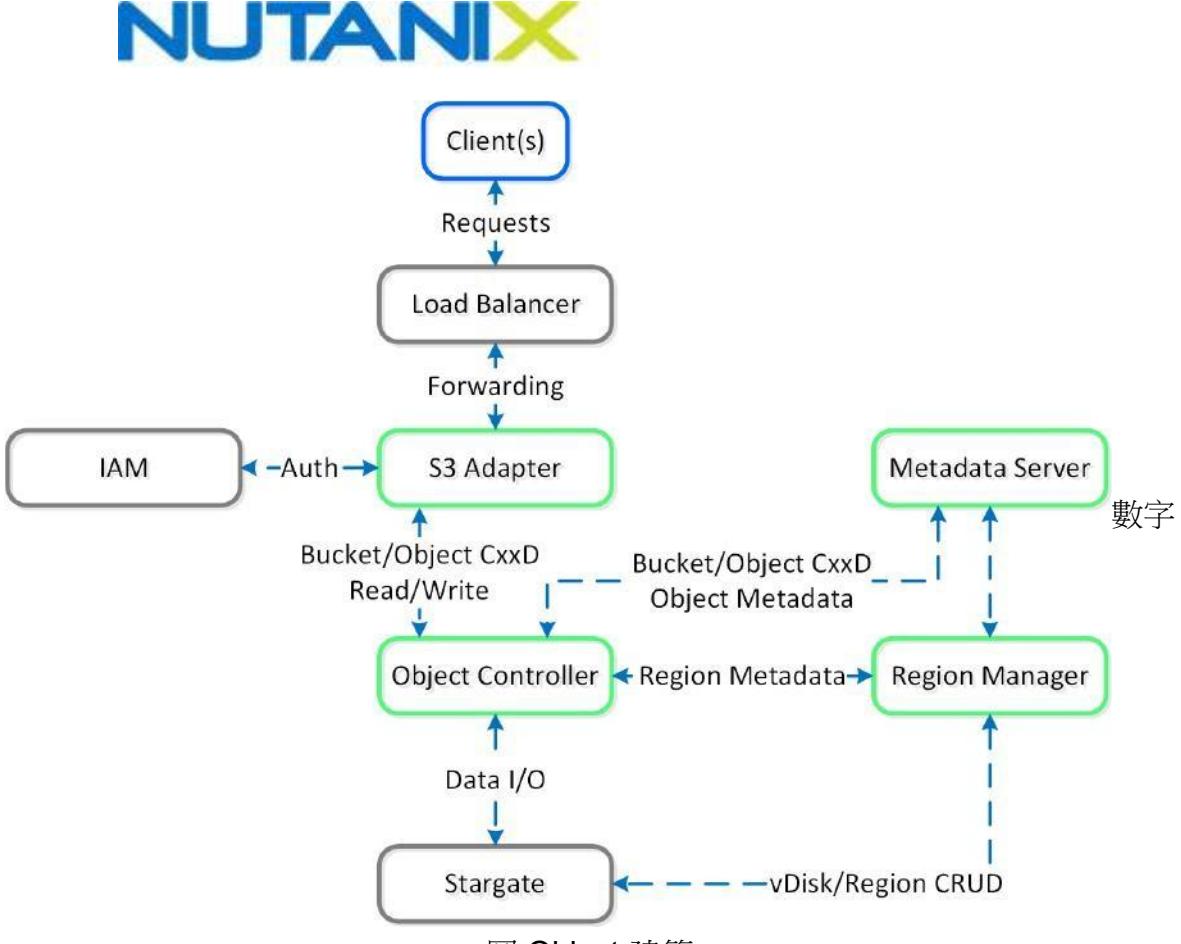


圖.Object-建築

Objects 特定的元件以 Nutanix 綠色突出顯示。對於 **Objects**，沒有“覆蓋”的概念，因此 **CxD** 與 **CRUD**（創建/替換/更新/刪除）不相關。物件“覆蓋”的常用方法是創建新修訂版或創建新物件並指向新物件。

Object 存儲和 I/O

Object 存儲在稱為區域的邏輯結構中。區域是虛擬磁片上固定的空間段。

該圖顯示了虛擬磁片與區域之間的關係的示例：

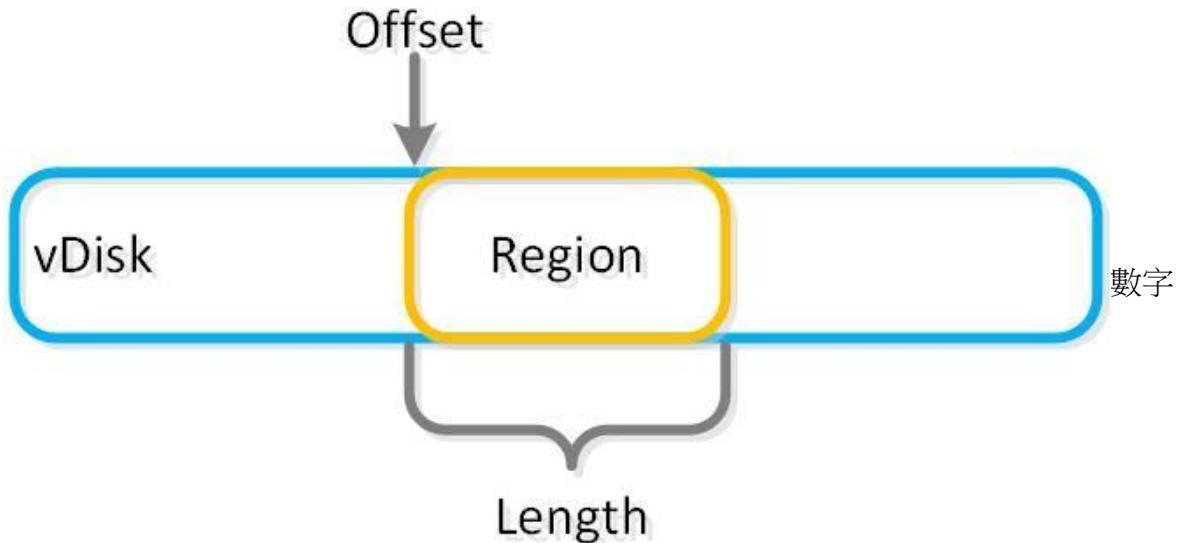


圖.Objects-vDisk Region

較小的 **Objects** 可能適合單個區域（區域 ID，偏移量，長度）的塊，而較大的 **Objects** 可能會在各個區域上出現條紋。當大型 **Objects** 跨多個區域劃分條帶時，這些區域可以託管在多個 **vDisk** 上，從而可以同時利用多個 **Stargate**。

該圖顯示了 **Object**，**chunk** 和 **region** 之間的關係的示例：

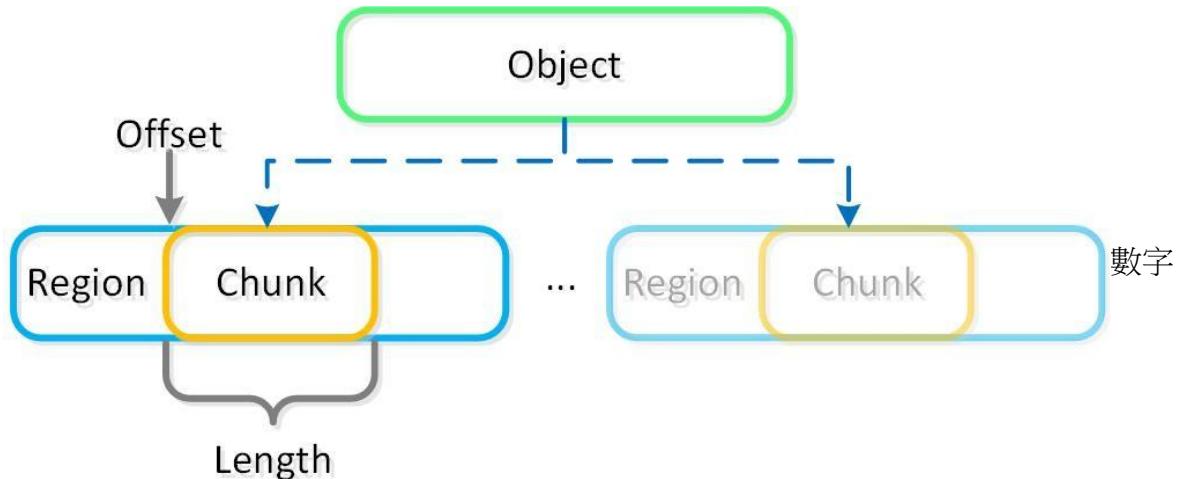


圖.物件-物件塊

Object 服務功能採用與 Nutanix 平臺相同的分發方法，以確保可用性和可擴展性。作為 **Objects** 部署的一部分，將至少部署 3 個 **Objects VM**。



10 第十部分：網路服務

10.1 Flow (微分段)

Flow 是一種分散式狀態防火牆，可在 AHV 平臺上運行的實體以及與其通信的外部事物之間進行精細的網路監視和實施。

支援的配置

該解決方案適用於以下配置（清單可能不完整，請參閱文檔以獲取完全支持的列表）：

核心用例：

- 微分段

管理介面：

- Prism Central (PC)

支援的環境：

- 本地：
 - AHV (從 AOS 5.10 開始)
- 雲：
 -

升級：

- AOS 的一部分

相容功能：

- Service Chaining
- Calm

Epoch

通過 Prism Central 定義策略並分配類別來完成配置。這允許在中央位置進行配置，並將其推送到許多 Nutanix 群集。每個 AHV 主機都使用 OpenFlow 實施規則。

實現結構

在 Nutanix Flow 中，有一些關鍵的結構：

類別

類別用於定義要對其應用策略和實施的實體組。它們通常適用，但不限於：環境，應用程式類型，應用程式層等。

- 類別：Key/Value “Tag”
- 示例：app app | tier | group | location | subnet 等等

例如，提供生產資料庫服務的 VM 可能具有以下分配的類別：

- AppTier：資料庫
- AppType：MySQL
- Environment：生產

然後，策略可以利用這些類別來決定要應用哪些規則/動作（也可以在 Flow 上下文之外利用）。

安全規則

安全規則是已定義的規則，並確定已定義類別之間允許的內容。



圖.Flow - Microsegmentation - Rules

有幾種類型的安全規則：

- 應用規則
 - 這是通用規則，可讓您定義允許/拒絕的傳輸方式（TCP / UDP），埠和源/目的地。
 - [允許|拒絕]傳輸：埠[至]寄件者]
 - 示例：允許 TCP 8080 從 Category : Tier : Web 到 Category : Tier : App
- 隔離規則
 - 拒絕兩個類別之間的流量，允許類別內的流量
 - 示例：將租戶 A 與租戶 B 分開，克隆環境並允許在不影響正常網路通信的情況下並行運行。
- 隔離規則
 - 拒絕指定 VM /類別的所有流量
 - 例如：VM A, B, C 感染了病毒，請將其隔離以阻止病毒進一步感染網路

下面顯示了一個示例，該示例利用 Flow-微分段來控制示例應用程式中的流量：

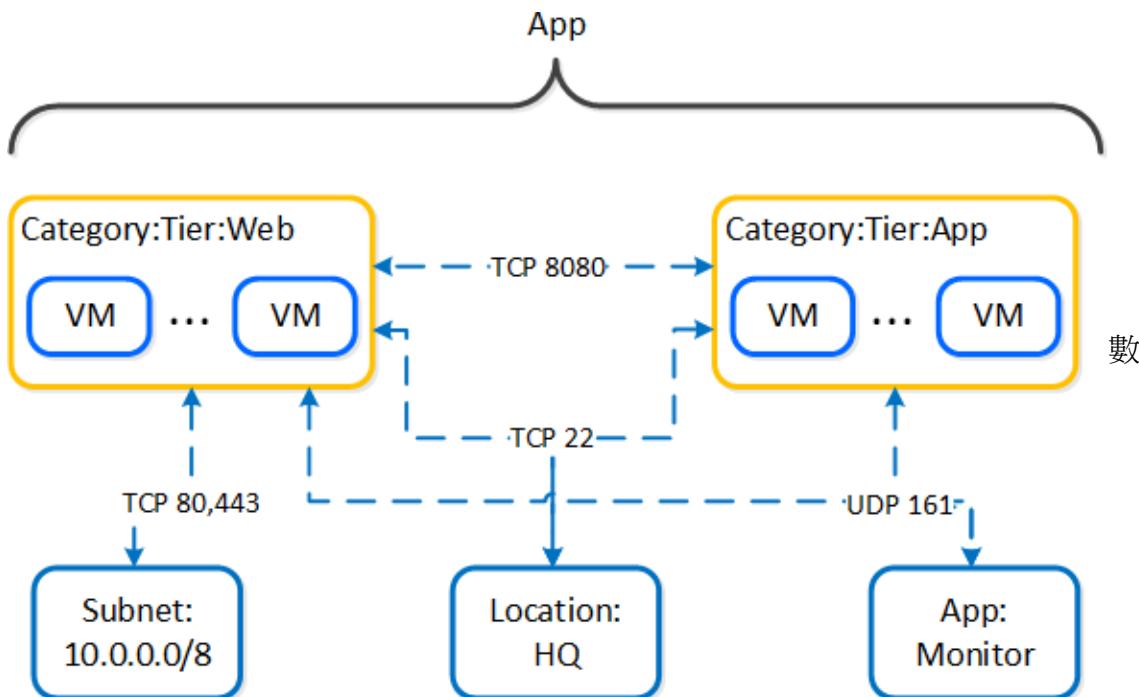


圖.Flow-微分段-示例應用

執行

執行決定匹配規則時將採取什麼措施。使用 AHV Flow-微分段，有兩種執行類型：

- 應用
 - 通過允許定義的流並丟棄所有其他流來實施策略。
- 監控
 - 允許所有流，但是在策略視覺化頁面中突出顯示任何可能違反該策略的資料包。

Flow-微分段規則在資料包離開 UVM 後首先應用于該資料包。這發生在微分段橋 (br.microseg) 中：

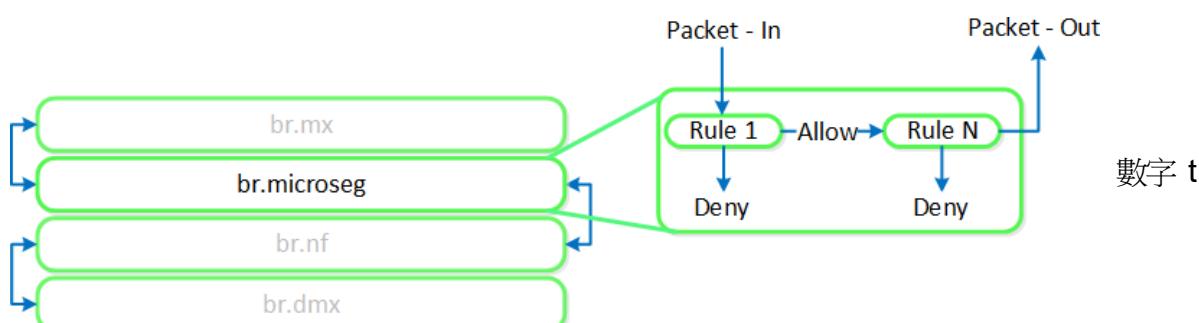


圖.流量-微分段-流量



10.2 Epoch (網路監控/ DPI)

待續

11 第十一部分：備份容災服務

11.1 Leap (策略驅動的災難恢復/運行手冊)

Nutanix Leap 功能提供了通過 Prism Central (PC) 配置的策略驅動的備份，災難恢復和運行手冊自動化服務。該功能建立在 AOS 上多年來在 PE 中配置的本機 DR 和複製功能之上，並對其進行了擴展。有關用於複製等的實際後端機制的更多資訊，請參閱“Acropolis”中的“備份和災難恢復 (DR)”部分。Leap 在 AOS 5.10 中引入。

支援的配置

該解決方案適用於以下配置（清單可能不完整，請參閱文檔以獲取完全支持的列表）：

核心用例：

- 基於策略的備份和複製
- DR 運行手冊自動化
- DRaaS (通過 Xi)

管理介面：

- Prism Central (PC)

支援的環境：

- 本地：
 - AHV (自 AOS 5.10 開始)
- 雲：
 - Xi (自 AOS 5.10 起)

升級：

- AOS 的一部分

相容功能：

- AOS BC / DR 功能

關鍵內容



在本節中將使用以下關鍵術語，術語定義如下：

•復原點目標（RPO）

- 指發生故障時可接受的資料丟失。例如，如果您希望 RPO 為 1 小時，則每 1 小時創建一次快照。如果進行還原，則您最多可以還原 1 個小時前的資料。對於同步複製，通常 RPO 為 0。

•恢復時間目標（RTO）

- 恢復時間目標。指從故障事件到恢復服務的時間段。例如，如果發生故障，並且您需要在 30 分鐘內備份和運行應用，則 RTO 為 30 分鐘。

•復原點

- 復原點又稱為快照。

實現構造

在 Nutanix Leap 中，有一些關鍵的結構：：

保護政策

關鍵角色：分配類別的備份/複寫原則

•說明：保護策略定義了 RPO（快照頻率），恢復位置（遠端群集/ Xi），快照保留（本地與遠端群集）以及相關的類別。使用保護策略，所有內容都將應用到類別級別（預設值可以應用於任何/全部）。這與必須選擇 VM 的保護域不同。

下圖顯示了 Nutanix 跳躍保護策略的結構：

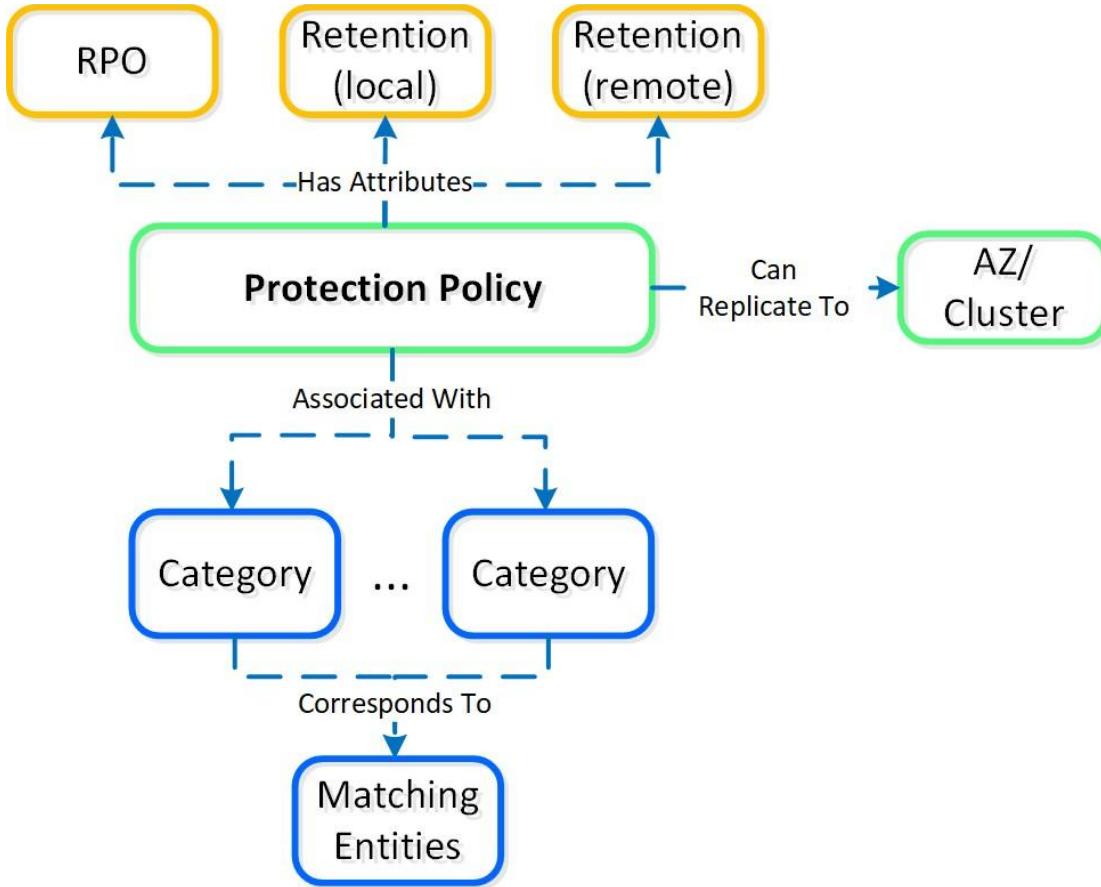


圖. Leap -保護策略

恢復計畫

- 關鍵角色：災難恢復手冊
- 說明：恢復計畫是一本運行手冊，它定義了開機順序（可以指定類別或 VM）和網路映射（主要與恢復以及測試容錯移轉/錯誤後回復）。這是人們對 SRM 的最多的同義詞定義。注意：必須先配置保護策略，然後才能配置恢復計畫。這是必需的，因為資料必須存在於恢復網站才能被恢復。

下圖顯示了 Nutanix Leap Recovery Plan 的結構：

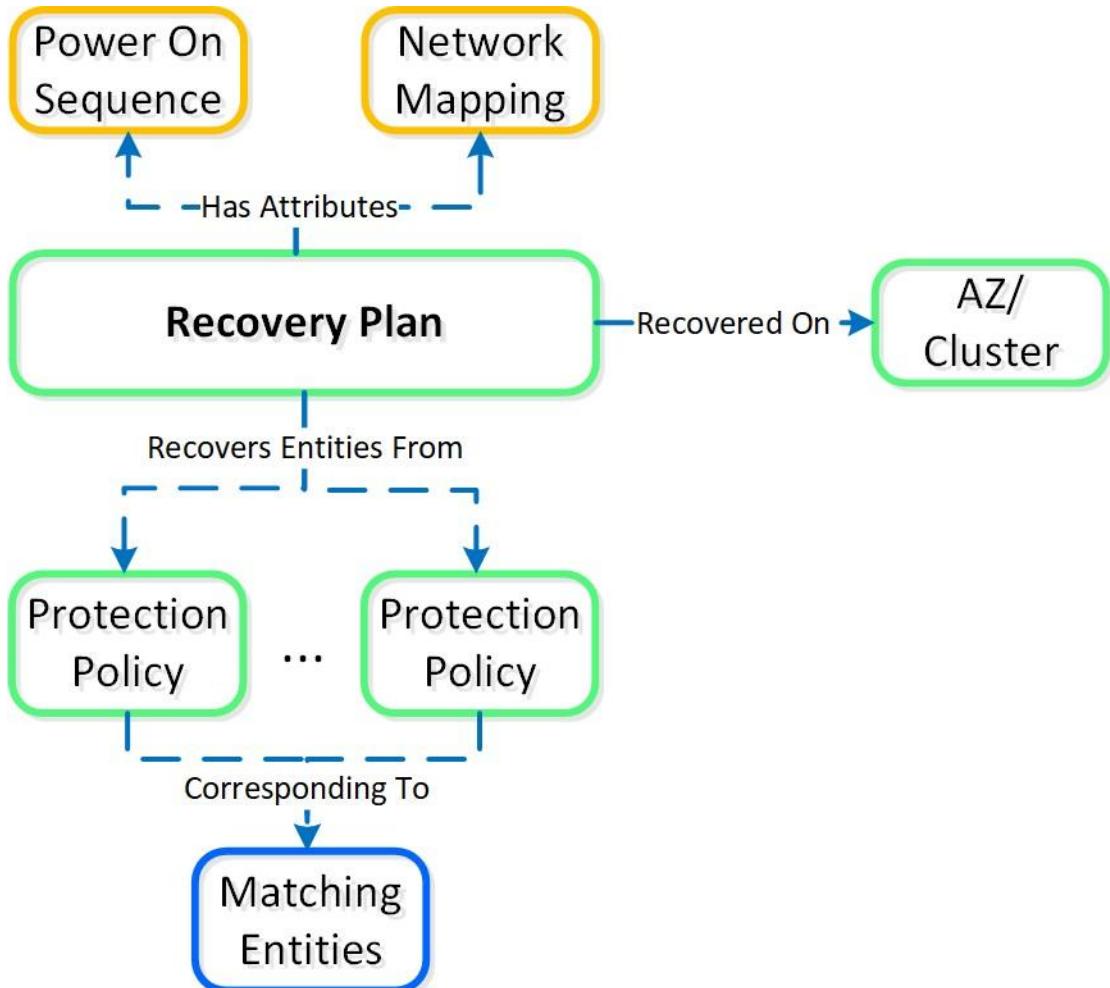


圖. Leap – 恢復計畫

線性保留政策

- 關鍵角色：復原點保留策略
- 描述：線性保留策略指定要保留的復原點的數量。例如，如果 RPO 為 1 小時，而您的保留時間設置為 10，則您將保留 10 小時 (10×1 小時) 的復原點（快照）。

匯總保留政策

- 關鍵角色：復原點保留策略
- 描述：匯總保留策略將根據 RPO 和保留持續時間“匯總”快照。例如，如果 RPO 為 1 小時，而您的保留時間設置為 5 天，則它將保持每小時 1 天和每天 4 天的復原點。該邏輯的特徵如下：如果保留時間為 n 天，則保留 1 天的 RPO 和 $n-1$ 天的每日恢復點。如果保留時間為 n 周，則保留 1 天的 RPO，每天保留 1 周，每週復原點保留 $n-1$ 周。如果保留時間為 n 個月，則保留 RPO 1 天，每天 1 周，每週 1 個月和每月 $n-1$ 個月的復原點。如果保留時間為 n 年，則保留 1 天的 RPO，每天保留 1 周，每週保留 1 個月，每月復原點保留 $n-1$ 個月。

線性與匯總保留

對於保留期較短的小型 RPO 視窗，或者在始終需要恢復到特定 RPO 視窗的情況下，請使用線性策略。

將匯總策略用於保留期較長的任何內容。它們更加靈活，可以自動處理快照的時效/精簡，同時第一天仍提供精細的 RPO。

下面顯示了 Leap 構造的高級概述：

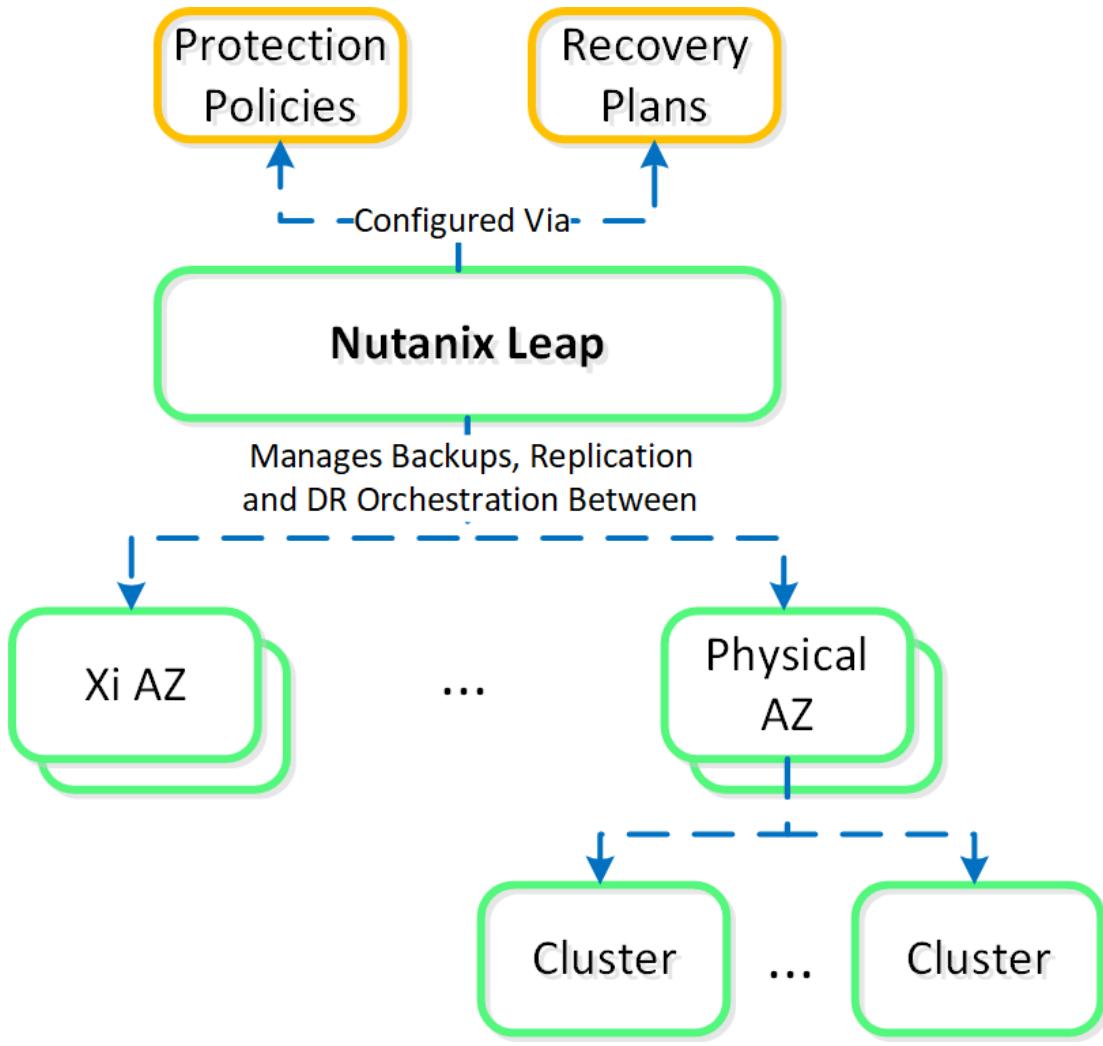


圖. Leap – 概覽

下面顯示了 Leap 如何在內部和 Xi 之間複製：

Leap can replicate subsets of VMs to Xi or other on-premise clusters.

In the event of a failure, Leap can failover VMs to both Xi and on-premise targets based on the policy.

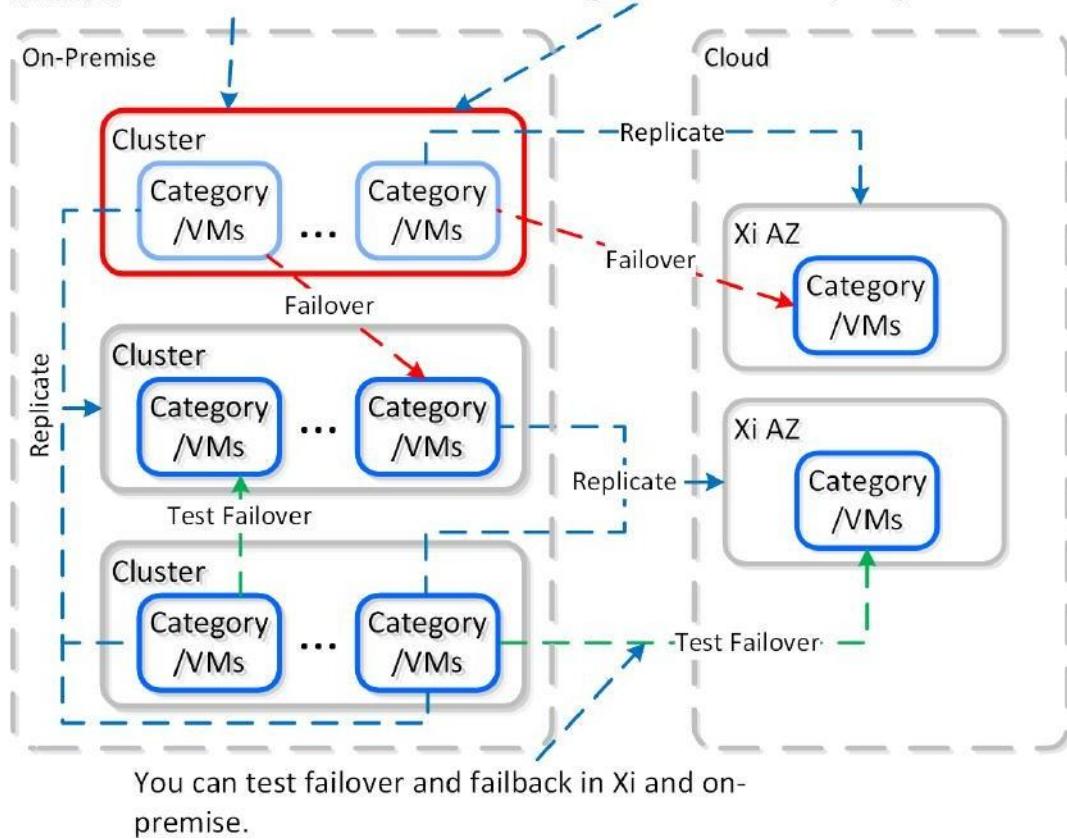


圖. Leap – 拓撲

用法和配置

以下各節介紹如何配置和使用 Leap。

總體過程可以分為以下步驟：

1. 連接到可用區 (AZ)
2. 配置保護策略
3. 配置恢復計畫
4. 執行/測試容錯移轉和錯誤後回復

連接可用區

第一步是連接到 Xi AZ 或另一台 PC 的 AZ。注意：從 5.11 開始，您至少需要部署 2 台 PC（每個網站 1 台）。

在 PC 中，搜尋“可用區域”或導航到“管理”->“可用區域”：



圖. Leap – 連接可用區

按一下“連接到可用區”，然後選擇 AZ 類型（“Xi”或“Physical Location”，又名 PC 實例）：

圖. Leap – 連接可用區

輸入 PC 或 Xi 的密碼，然後按一下“連接”：



Connect to Availability Zone

?

X

Availability Zone Type

Physical Location

IP Address for Remote PC

99.99.99.99

Username

foobar

Password

.....|

Close

Connect

圖. Leap – 連接可用區

現在將會顯示已連接的可用區並可用。

配置保護策略

在 PC 中，搜索“保護策略”或導航到“策略”->“保護策略”：

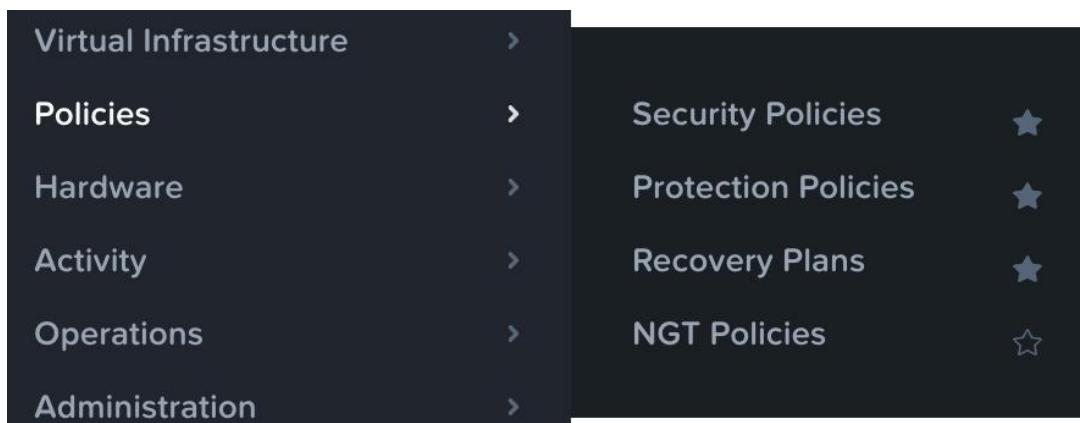


圖. Leap – 保護策略



點擊“創建保護政策”：

The screenshot shows the Nutanix Leap interface with the title "Protection Policies". It displays a message "0 Total Protection Policies". There is a search bar labeled "Type name to filter by" and a blue button labeled "Create Protection Policy". A dropdown menu icon is also visible.

圖. Leap – 創建保護策略

輸入名稱，恢復位置，RPO 和保留策略的詳細資訊（如前所述）：



Name

myProtectionPolicy

Primary Location

Local AZ

Recovery Location

Target Cluster ?

PC_10.47.17.10

EARTH-DEV-1

When VMs failover to recovery location, reverse replication to the primary location will be initiated automatically with this same Protection Policy.

Recovery Point Objective

Start immediately [Change](#)

Hours

1

Retention Policy

Linear

Roll-up

Remote Retention

-

6

+

Months

▼

24 hourly, 7 daily, 4 weekly, 6 monthly recovery points will be retained.

Local Retention

-

5

+

Days

▼

24 hourly, 5 daily recovery points will be retained.

Take App-Consistent Recovery Point ?

圖. Leap – 保護策略輸入

注意：對於 Xi，您不需要選擇“目標集群”



Name

MyXiProtectionPolicy

Primary Location

Local AZ

Recovery Location

Target Cluster

US-EAST-1B

autoselect

When VMs failover to recovery location, reverse replication to the primary location will be initiated automatically with this same Protection Policy.

To replicate to Xi, certain ports on your firewall may need to be opened. [Learn More](#)

Recovery Point Objective

Start immediately [Change](#)

Hours

1

Retention Policy

Linear

Roll-up

Remote Retention

-

1

+

Years

▼

24 hourly, 7 daily, 4 weekly, 12 monthly, 1 yearly recovery points will be retained.

Local Retention

-

5

+

Days

▼

24 hourly, 5 daily recovery points will be retained.

Current pricing level: **Premium**

[View Pricing Detail](#)

Take App-Consistent Recovery Point



圖. Leap – 保護策略輸入 – Xi

接下來，我們將選擇適用於該政策的類別：

Add Categories X

Select Categories

AppType:Earth_Stack +

- Environment:Production

Close Save

圖. Leap – 保護策略類別

按一下“保存”，您將看到新創建的保護策略：

1 Total Protection Policies						
<input type="checkbox"/>	Name	Primary Location	Recovery Location	RPO	Remote Retention	Local Retention
<input type="checkbox"/>	myProtectionPolicy	Local AZ	PC_10.471710	1 hour(s)	6 Month(s)	5 Day(s)

圖. Leap – 保護策略

配置恢復計畫

在 PC 中，搜索“恢復計畫”或導航到“策略”->“恢復計畫”：



The screenshot shows the Nutanix LEAP interface. On the left, there's a vertical navigation bar with the following items: Virtual Infrastructure, Policies, Hardware, Activity, Operations, and Administration. To the right of this is a dark sidebar containing four items: Security Policies (marked with a star), Protection Policies (marked with a star), Recovery Plans (marked with a star), and NGT Policies (marked with a star).

圖. Leap – 恢復計畫

在首次啟動時，將看到用於創建第一個恢復計畫的螢幕：

This screenshot shows the initial screen for creating a recovery plan in Nutanix LEAP. It features a central diagram illustrating a failover scenario between two locations. Below the diagram, the text "Introducing Recovery Plans" is displayed, followed by a sub-instruction: "Use Recovery Plans to configure power on sequence and network settings for failover". A prominent blue button labeled "Create New Recovery Plan" is centered at the bottom.

Recommended steps before creating the Recovery Plan

Enable Disaster Recovery	Setup Networks	Protect VMs
Enable Leap on the Primary and Recovery Location	Setup networks between Primary & Recovery location for failover	Ensure VMs to be part of a Recovery Plan are protected to a unique recovery location

圖. Leap – 創建恢復計畫

使用下拉式功能表選擇“恢復位置”：



Select Location

X

Primary Location

Local AZ



Recovery Location

PC_10.47.17.10



Close

Proceed

圖. Leap – 選擇恢復位置

注意：可以是任一 Xi 可用區或物理可用區（具有相應託管群集的 PC）。
輸入恢復計畫名稱和描述，然後按一下“下一步”：



1 General 2 Power On Sequence 3 Network Settings

Recovery Plan Name

myRecoveryPlan

Recovery Plan Description

Recovery Plan Description

Next

圖. Leap – 恢復計畫 – 命名

接下來點擊“添加實體”並指定開機順序：



1

General

2

Power On Sequence

3

Network Settings

Start by adding Entities to your Power On Sequence.

+ Add Entities

◀ Back

Next

圖 Leap – 恢復計畫 – 啟動循序

搜尋要添加到每個階段的 VM 或類別：



Add Entities

X

Search Entities by

VM Name	▼	Enter Search Text	
VM Name			
Category			

Close

Add

圖 Leap – 恢復計畫 – 啟動順序

一旦各階段的開機順序看起來不錯，請按一下“下一步”：



1 General 2 Power On Sequence 3 Network Settings

3 Stages | 3 Total | 2 Categories, 1 VMs

+ Add New Stage

STAGE 1 1 Total | 1 VMs

Actions ↓ ↑ -

Name

prod-earth-db

+ Add Delay

STAGE 2 1 Total | 1 Categories

Actions ↓ ↑ -

Name

CATEGORY AppTier DB

+ Add Delay

STAGE 3 1 Total | 1 Categories

Actions ↓ ↑ -

Name

CATEGORY AppTier Kafka

Back

Next

圖 Leap – 恢復計畫 – 啟動順序

啟動順序



確定開機順序時，您將需要分階段進行以下操作：

- 階段 0：核心服務（AD，DNS 等）
- 階段 1：取決於階段 0 服務的服務，是階段 2 服務所必需的（例如資料庫層）
- 階段 2：服務取決於階段 1 服務，是階段 3 服務所必需的（例如應用層）
- 階段 3：服務取決於階段 2 服務，並且是階段 4 服務所必需的（例如 Web 層）
- 階段 4-N：根據依賴性重複

現在，我們將在源環境和目標環境之間映射網路：

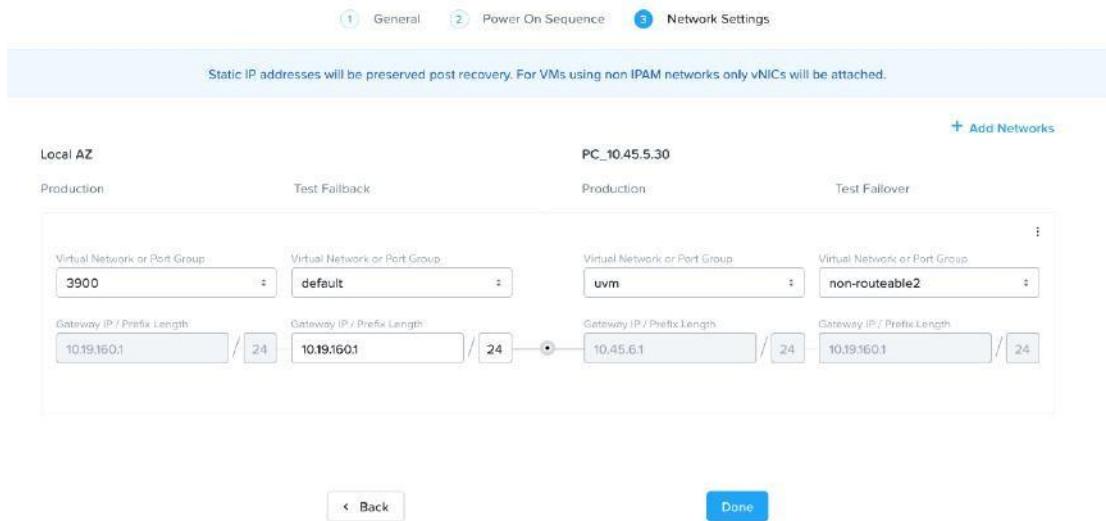


圖. Leap – 恢復計畫 – 網路映射

容錯移轉/錯誤後回復網路

在大多數情況下，您將需要對測試網路使用不可路由或隔離的網路。這將確保您沒有重複 SID，arp 條目等問題。

11.2 Mine (備份解決方案)

待續

12 第十二部分：編排服務

12.1 Calm (編排/自動化)

待續



13 第十三部分：治理服務

13.1 Beam (成本治理/合規性)

待續

14 第十四部分：場景

14.1 場景：安全分析平臺

待續

14.2 場景：多網站災難恢復/複製

待續

15 第十五部分：集成

15.1 集成

15.1.1 OpenStack

OpenStack 是一個用於管理和構建雲的開源平臺。它主要被分為前端(儀錶盤和 API)和基礎架構服務(計算，存儲資源等)。

OpenStack 和 Nutanix 解決方案有以下主要組件構成：

- OpenStack 控制器(OSC)
 - 一個現存的或新創建的虛擬機器或主機，用來提供 OpenStack 使用者介面、API 和服務。處理所有 OpenStack 的 API 調用。它可以跟 Acropolis OpenStack 驅動共存在同一個 Acropolis OVM 中。
- Acropolis OpenStack 驅動

- 負責處理來自 OpenStack 控制器的 OpenStack RPC，並將其轉換為本地 Acropolis API 調用。它可以部署在 OpenStack 控制器，預裝的 OVM 或是一個新的虛擬機器上。
- Acropolis OpenStack 服務虛擬機器 (OVM)
 - 具有 Acropolis 驅動的虛擬機器，負責處理來自 OpenStack 控制器的 OpenStack RPC，並將其轉換為本地 Acropolis API 調用。

OpenStack 控制器可以是一個已經存在的虛擬機器/主機，或者作為 OpenStack 的一部分部署在 Nutanix 解決方案上。Acropolis OVM 是一個助手 VM，它作為 Nutanix OpenStack 解決方案的一個部分部署。

用戶端用他們期望的方式 (Web UI / HTTP, SDK, CLI or API) 來與 OpenStack 控制器進行通信。而 OpenStack 控制器則與 Acropolis OVM 通信，後者利用 OpenStack 驅動把指令轉換成 Acropolis 自身的 REST API 調用。

下圖描述這種通訊的概要過程：

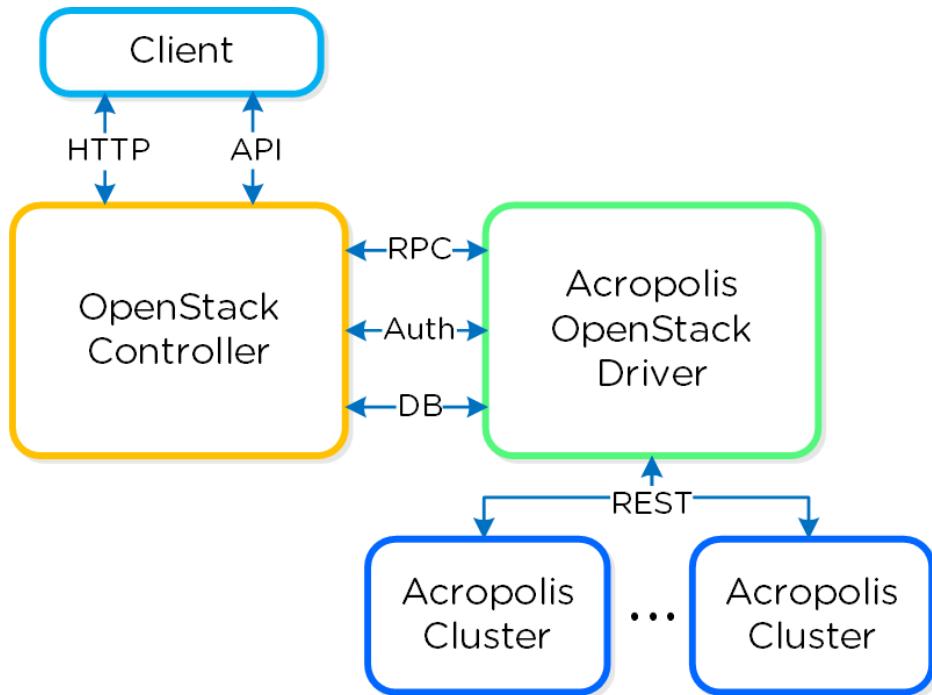


圖. OpenStack + Acropolis OpenStack 驅動

這樣既可以使兩者完美結合，也可以實現 OpenStack Portal 和 API 的優點，而無需複雜的 OpenStack 基礎架構和相關管理。所有後端基礎架構服務（計算，存儲，網路）均利用本地 Nutanix 服務。無需部署 Nova Compute 主機等。該平臺公開了這些服務的 API、控制器與這些服務進行通信，然後將它們轉換為本地 Acropolis API 調用。此外，考慮到簡化的部署模型，完整的 OpenStack + Nutanix 解決方案可以在不到 30 分鐘的時間內部署完成。

支援的 OpenStack 控制器

當前版本 (對應於 4.5.1) 需要一個 Kilo 或更新版本的 OpenStack 控制器。

下表列出了各元件角色映射的概要資訊：

條目	角色	OpenStack 控制器	Acropolis OVM	Acropolis 集群	Prism
Tenant Dashboard	User interface and API	X			
Admin Dashboard	Infra monitoring and ops	X			X
Orchestration	Object CRUD and lifecycle management	X			
Quotas	Resource controls and limits	X			
Users, Groups and Roles	Role based access control (RBAC)	X			
SSO	Single-sign on	X			
Platform Integration	OpenStack to Nutanix integration		X		
Infrastructure Services	Target infrastructure (compute, storage, network)			X	

15.1.1.1 OpenStack 組件

OpenStack 由一系列提供基礎架構功能服務的元件構成。其中一些功能由 OpenStack 控制器提供，有些則由 Acropolis OVM 提供。

下表列出了核心的 OpenStack 組件及其角色的映射：

組件	角色	OpenStack 控制	Acropolis OVM

Keystone	Identity service	X	
Horizon	Dashboard and UI	X	
Nova	Compute		X
Swift	Object storage	X	X
Cinder	Block storage		X
Glance	Image service	X	X
Neutron	Networking		X
Heat	Orchestration	X	
Others	All other components	X	

下面提供了一個 OpenStack 元件及其通訊過程更為詳細的視圖：

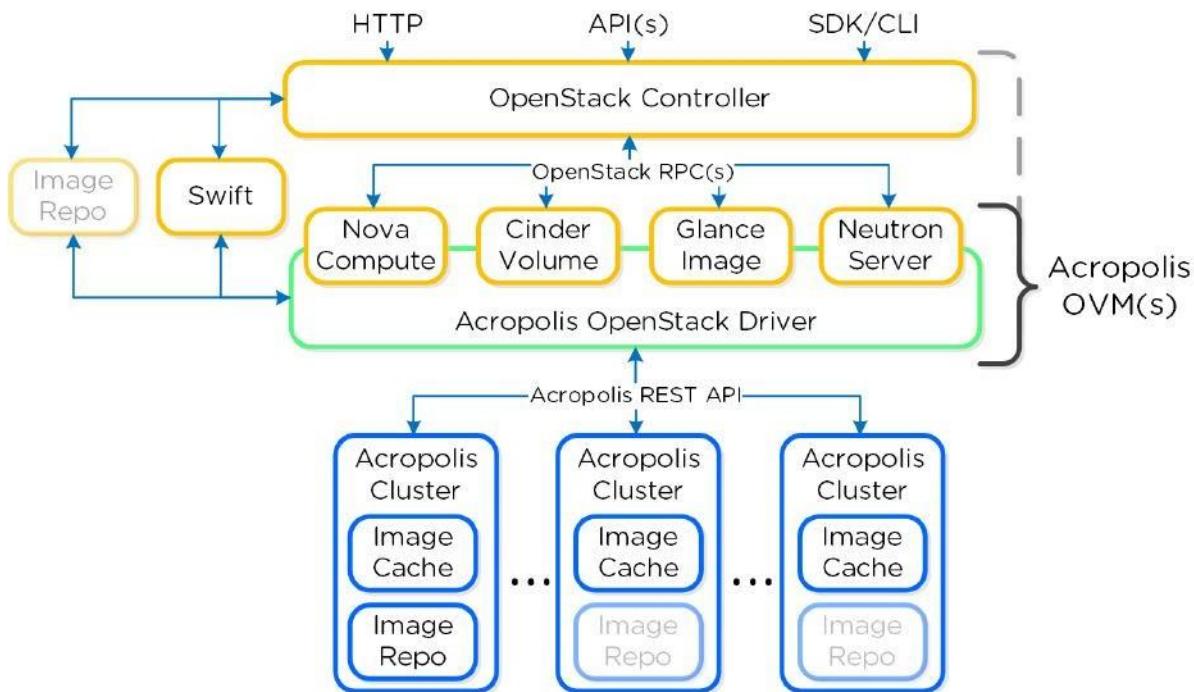


圖. OpenStack + Nutanix API 通訊

在下面的章節中，我們將討論一些主要的 OpenStack 組件，以及它們如何集成到 Nutanix 平臺中。

Nova



Nova 是 OpenStack 平臺的計算資源引擎和調度器。在 Nutanix OpenStack 解決方案中，每個 Acropolis OVM 作為一個計算主機，每個 Acropolis 集群都將作為一個單一的 hypervisor 主機，用於調度 OpenStack 實例。Acropolis OVM 運行 Nova 計算資源服務。

你可以通過 OpenStack 門戶從下面路徑中查看 Nova 服務：

'Admin'->'System'->'System Information'->'Compute Services'.

下圖展示了 Nova 服務，主機及其狀態：

System Information

Name	Host	Zone	Status	State	Last Updated
nova-consoleauth	OSCTRL01	internal	Enabled	Up	0 minutes
nova-scheduler	OSCTRL01	internal	Enabled	Up	0 minutes
nova-conductor	OSCTRL01	internal	Enabled	Up	0 minutes
nova-cert	OSCTRL01	internal	Enabled	Up	0 minutes
nova-compute	ASVM01	US-West-1	Enabled	Up	0 minutes
nova-compute	ASVM02	US-West-1	Enabled	Up	0 minutes
nova-compute	ASVM03	US-West-2	Enabled	Up	0 minutes

圖. OpenStack Nova 服務

Nova 調度器基於選定的可用區，決定在哪個計算主機（如 Acropolis OVM）上存放實例。這些請求將會被發送到選定的 Acropolis OVM 上，然後由 Acropolis OVM 轉發請求到目標主機的 Acropolis 調度器。Acropolis 調度器將在集群中選出最優的節點存放方式。集群中各節點不會直接暴露給 OpenStack。

你可以通過 OpenStack 門戶，在路徑'Admin'->'System'->'Hypervisors'中查看電腦和 hypervisor 主機。

下圖以電腦主機的形式列出了 Acropolis OVM：

Hypervisor	Compute Host

Host	Zone	Status	State
ASVM01	US-West-1	enabled	up
ASVM02	US-West-1	enabled	up
ASVM03	US-West-2	enabled	up

圖. OpenStack Compute 主機

下圖以 hypervisor 主機的形式列出了 Acropolis 集群：



Hypervisor Compute Host

Hostname	Type	VCPUs (used)	VCPUs (total)	RAM (used)	RAM (total)	Local Storage (used)	Local Storage (total)	Instances
TMBEAST	Acropolis	4	448	8.5GB	1.7TB	80GB	25.9TB	1

圖. OpenStack Hypervisor 主機
如你從上面的圖示所見，所有集群的資源以單個 hypervisor 主機的方式展現。

Swift

在物件存儲中 Swift 被用於存儲和檢索檔。目前它僅用於快照和圖像的備份/恢復。

Cinder

Cinder 是 OpenStack 的卷元件，對外提供 iSCSI 物件。在 Nutanix 解決方案中 Cinder 的實現借助於 Acropolis 卷管理 API。這些卷將以塊設備的形式直接附加到實例上去（相較於 in-guest）。

你可以通過 OpenStack 門戶，在路徑'Admin'->'System'->'System Information'->'Block Storage Services'下查看 Cinder 服務。

下圖顯示了 Cinder 服務，主機和狀態：

System Information

Services Compute Services Block Storage Services Network Agents Orchestration Services

Filter

Name	Host	Zone	Status	State	Last Updated
cinder-backup	OSCTRL01	nova	Enabled	Up	0 minutes
cinder-scheduler	OSCTRL01	nova	Enabled	Up	0 minutes
cinder-volume	OSCTRL01@lvm	nova	Enabled	Down	1 day, 23 hours
cinder-volume	ASVM01@acropolis	nova	Enabled	Up	0 minutes
cinder-volume	ASVM02@acropolis	nova	Enabled	Up	0 minutes
cinder-volume	ASVM03@acropolis	nova	Enabled	Up	0 minutes

Displaying 6 items

圖. OpenStack Cinder 服務

Glance / Image Repo

Glance 是 OpenStack 中的鏡像庫，展現部署實例的可用鏡像。鏡像包括 ISO，磁片和快照。

Image Repo 是用來保存由 Glance 發佈的可用鏡像的存儲庫。這些鏡像可以位於 Nutanix 環境中或由外部來源。當鏡像被保存在 Nutanix 平臺上時，他們通過 OVM 以 Glance 的形式發佈到 OpenStack 控制器。當 Image Repo 只存在于外部源時，Glance 將被 OpenStack 控制器接管，並且 Image Cache 將在 Acropolis Cluster 上使用。

Glance 在每個群集的基礎上啟用，並且總是和 Image Repo 共存。在多個集群中同時啟動 Glance 時，Image Repo 會橫跨這些集群，並且通過 OpenStack 們創建的鏡像將會被推送到所有運行 Glance 的集群。沒有運行 Glance 的集群將利用 Image Cache 在本地緩存鏡像。



專家提示

對於大規模部署，每個網站至少保證有兩個 Acropolis 集群在運行 Glance。這樣就能對 Image Repo 實現高可用，以便在一個集群不可用時，鏡像始終可用，即便它不在 Image Cache 中。

當以外部源運行 Image Repo/Glance 時，Nova 會負責把資料從外部源轉移到目標端的 Acropolis 集群。在這種情況下，目標端的 Acropolis 集群會利用 Image Cache 來緩存鏡像，以便後續訪問時能從本地讀取。

Neutron

Neutron 是 OpenStack 中的網路元件，負責網路配置。Acropolis OVM 允許使用者通過OpenStack 門戶進行網路 CRUD 的操作，然後在 Acropolis 中進行必要的修改。
你可以通過 OpenStack 門戶，從'Admin'->'System'->'System Information'->'Network Agents'查看 Neutron 服務。

下圖展示了 Neutron 服務，主機和狀態：

System Information

Type	Name	Host	Status	State	Last Updated
L3 agent	neutron-l3-agent	ASVM01	Enabled	Up	0 minutes
Metadata agent	neutron-metadata-agent	ASVM01	Enabled	Up	0 minutes
DHCP agent	neutron-dhcp-agent	ASVM01	Enabled	Up	0 minutes
Open vSwitch agent	neutron-openvswitch-agent	ASVM01	Enabled	Up	0 minutes

+ 目前只支援 Local 和 VLAN 網路類型

圖. OpenStack Neutron 服務

Neutron 在實例啟動時分配 IP 位址。Acropolis 也正是以這種方式得到相應虛擬機器的IP 位址。當虛擬機器發出 DHCP 請求時，Acropolis Master 將像對待 Acropolis Hypervisor 一樣在專用 VXLAN 上對此請求做出回應。

被支援的網路類型

當前只支援 Local 和 VLAN 網路類型

Keystone 和 Horizon 元件運行在與 Acropolis OVM 介面的 OpenStack Controller 中。OVM 中的 OpenStack 驅動將把 OpenStack API 調用轉換成本地的 Acropolis API 調用。

15.1.1.2 設計和部署



對於大規模的雲部署，一個好的交付拓撲是至關重要的。它將被分發並滿足最終用戶的需求，同時提供靈活性和本地性。

下面是在 OpenStack 中用到的高層級的概念：

- 區域-Region
 - 一個地理意義上區域，其上可構建多個可用區（AZ）。區域可以設置成像 US-Northwest 或是 US-West 這樣。
- 可用區-Availability Zone (AZ)
 - 一個運行著雲服務的特定網站或資料中心的位置。它可以包含像是 US-Northwest-1 or US-West-1 這樣的網站。
- 主機聚合-Host Aggregate
 - 一組計算主機，可以是一排、一整片或是如網站或 AZ 那樣的範圍。
- 計算主機-Compute Host
 - 一個運行著 nova 計算服務的 Acropolis OVM。
- 虛擬管理程式主機-Hypervisor Host
 - 一個 Acropolis 集群（以一個單一主機的形式可見）。

下圖展示了以上各概念的高層級關係：

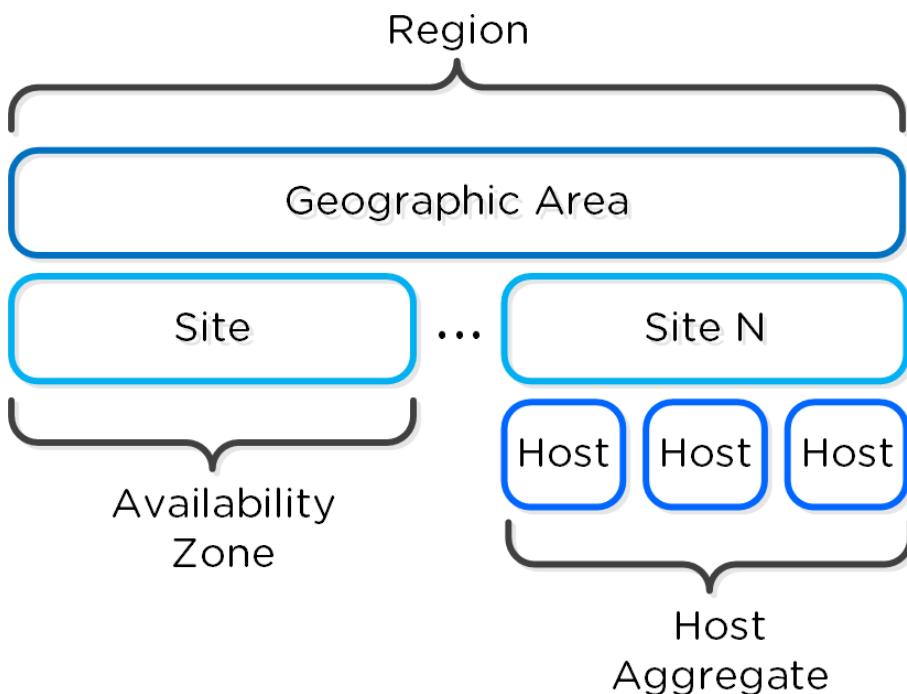


圖. OpenStack - 部署結構



下圖展示了各概念應用的例子：

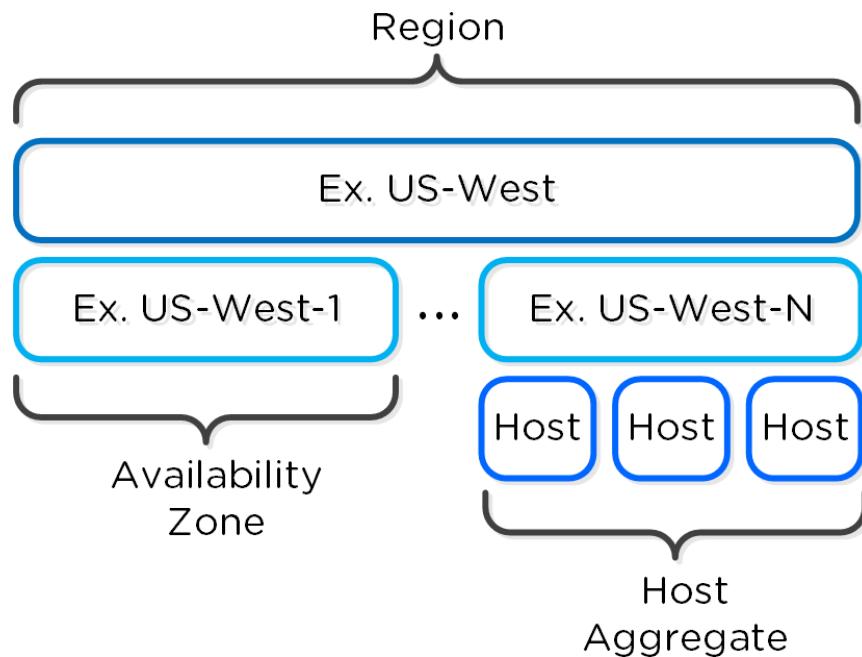


圖. OpenStack - Deployment Layout - 實例

你可以通過 OpenStack 門戶，從'Admin'->'System'->'Host Aggregates'查看和管理主機，主機聚合和可用區。

下圖展示了主機聚合，可用區和主機：

Host Aggregates

Host Aggregates					<input type="button" value="Filter"/> <input type="button" value="Create Host Aggregate"/> <input type="button" value="Delete Host Aggregates"/>
	Name	Availability Zone	Hosts	Metadata	Actions
<input type="checkbox"/>	FOOCLU01	US-West-2	ASVM03	availability_zone = US-West-2	<input type="button" value="Edit Host Aggregate"/>
<input type="checkbox"/>	TMBEAST1	US-West-1	ASVM01 ASVM02	availability_zone = US-West-1	<input type="button" value="Edit Host Aggregate"/>
Displaying 2 items					

Availability Zones

Availability Zone Name	Hosts	Available
internal	OSCTRL01 (Services Up)	Yes
US-West-1	ASVM01 (Services Up) ASVM02 (Services Up)	Yes
US-West-2	ASVM03 (Services Up)	Yes
Displaying 3 items		

圖. OpenStack 主機聚合和可用區

15.1.1.3 服務設計和擴展

對於大規模部署來說，推薦以負載均衡的方式把多個 Acropolis OVM 連接到 OpenStack 控制器。這樣既保證了可用性又均衡了工作負載。OVM 本身不包含任何處於延展性要求的狀態資訊。

下圖展示了如何在一個單一網站擴展 OVM：

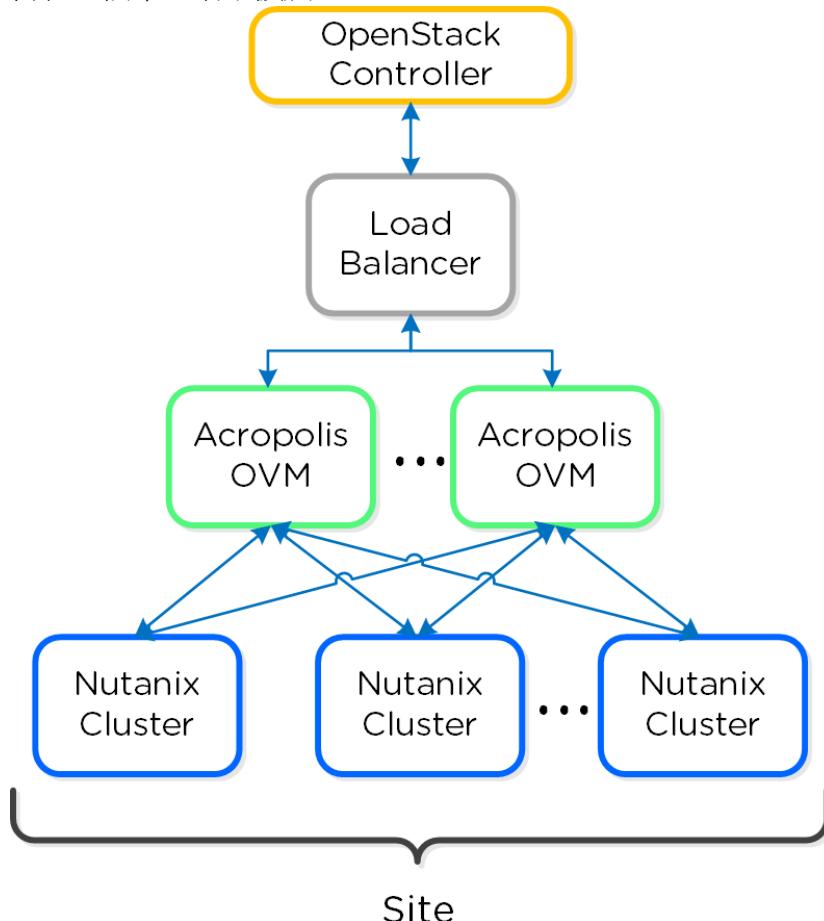


圖. OpenStack - OVM 負載均衡

實現 OVM 的一種方法是使用 Keepalived 和 HAProxy。

對於跨多個網站的環境，OpenStack 控制器將與跨網站的多個 Acropolis OVM 進行通信。

下圖展示了一個跨網站部署的例子：

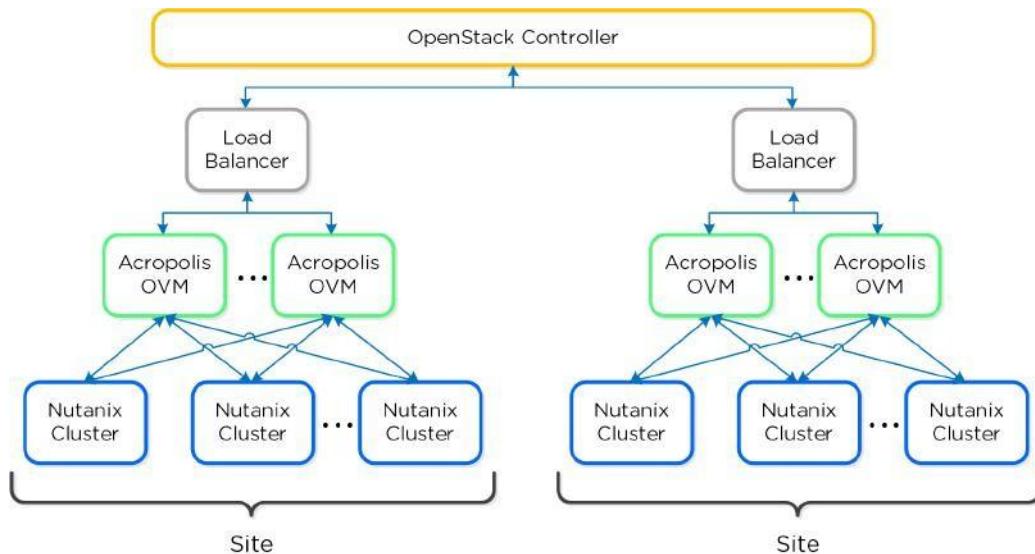


圖. OpenStack - 多網站

15.1.1.4 部署

OVM 可以以獨立的 RPM 包形式部署在 CentOS/Redhat 發行版本本當中，或是作為一個虛擬機器直接運行。Acropolis OVM 可以部署在任何 Nutanix 或非 Nutanix 的平臺上，只要其能連接到 OpenStack 控制器和 Nutanix 集群。

Acropolis OVM 的虛擬機器可以通過以下方式部署到 Nutanix AHV 集群上。如果 OVM 已經部署，您可以跳過 VM 創建步驟。你可以用完整的 OVM 鏡像或使用現有的 CentOS / Redhat VM 鏡像。

首先我們將把 Acropolis OVM 的磁片鏡像導入到 Acropolis 集群。可以用 SCP 來傳輸鏡像或是指定一個 URL 來導入。我們將用鏡像服務的 API 來導入。注意：OVM 的虛擬機器可以部署到任何地方，不一定是 Acropolis 集群。

運行以下命令，通過鏡像 API 導入磁片鏡像：

```
image.create <IMAGE_NAME> source_url=<SOURCE_URL> container
```

從任何 CVM 運行下列 ACCLI 命令來創建 OVM 虛擬機器：

```
vm.create <VM_NAME> num_vcpus=2 memory=16G
vm.disk_create <VM_NAME> clone_from_image=<IMAGE_NAME>
vm.nic_create <VM_NAME> network=<NETWORK_NAME>
vm.on <VM_NAME>
```

虛擬機器創建完畢並開啟後，用給定的用戶名和口令 SSH 到 OVM。



OVMCTL Help

在 OVM 上運行以下命令來獲得說明文本：

```
ovmctl --help
```

OVM 支援兩種部署方式：

- OVM-allinone
 - OVM 包含所有 Acropolis 驅動和 OpenStack 控制器
- OVM-services
 - OVM 包含所有 Acropolis 驅動並且和外部/遠端的 OpenStack 控制器通訊

這兩種部署模式都將在以下部分中進行討論。你可以在任何模式下使用，也可以在模式之間切換。

OVM-allinone

下列步驟涵蓋了 OVM-allinone 的部署，從 SSH 到 OVM 開始：

```
# Register OpenStack Driver service
ovmctl --add ovm --name <OVM_NAME> --ip <OVM_IP>

# Register OpenStack Controller
ovmctl --add controller --name <OVM_NAME> --ip <OVM_IP>

# Register Acropolis Cluster(s) (run for each cluster to add)
ovmctl --add cluster --name <CLUSTER_NAME> --ip <CLUSTER_IP> --username <PRISM_USER> --
password <PRISM_PASSWORD>
```

The following values are used as defaults:

Number of VCPUs per core = 4

Container name = default

Image cache = disabled, Image cache URL = None

我們可以用下面命令來確認配置：



```
ovmctl --show
```

到此為止，所有的服務都應該起來並且正常運行。

OVM-services

下列步驟介紹了 OVM-services 的部署方式，從 SSH 到 OVM 開始：

```
# Register OpenStack Driver service
ovmctl --add ovm --name <OVM_NAME> --ip <OVM_IP>

# Register OpenStack Controller
ovmctl --add controller --name <OS_CONTROLLER_NAME> --ip <OS_CONTROLLER_IP> --username
<OS_CONTROLLER_USERNAME> --password <OS_CONTROLLER_PASSWORD>
```

The following values are used as defaults:

Authentication: auth_strategy = keystone, auth_region = RegionOne
auth_tenant = services, auth_password = admin
Database: db_{nova,cinder,glance,neutron} = mysql, db_{nova,cinder,glance,neutron}_password = admin
RPC: rpc_backend = rabbit, rpc_username = guest, rpc_password = guest

```
# Register Acropolis Cluster(s) (run for each cluster to add)
ovmctl --add cluster --name <CLUSTER_NAME> --ip <CLUSTER_IP> --username <PRISM_USER> --
password <PRISM_PASSWORD>
```

The following values are used as defaults:

Number of VCPUs per core = 4
Container name = default
Image cache = disabled, Image cache URL = None

如果 OpenStack 控制器用的是非缺省的口令，我們需要更新以下資訊：

```
# Update controller passwords (if non-default are used)
ovmctl --update controller --name <OS_CONTROLLER_NAME> --auth_nova_password <> --
auth_glance_password <> --auth_neutron_password <> --auth_cinder_password <> --db_nova_password <> --
db_glance_password <> --db_neutron_password <> --db_cinder_password <>
```



接下來我們用下面命令來確認配置：

```
ovmctl --show
```

既然 OVM 已經被設置完畢，我們就要配置 OpenStack 控制器使其能夠獲知 Glance 和 Neutron 的 endpoint 信息。

登錄 OpenStack 控制器並進入 kestonerc_admin 源環境：

```
# enter kestonerc_admin source ./kestonerc_admin
```

首先我們要刪除原來指向控制器的 Glance endpoint：

```
# Find old Glance endpoint id (port 9292) keystone endpoint-list # Remove old keystone endpoint for  
Glance  
keystone endpoint-delete <GLANCE_ENDPOINT_ID>
```

下一步我們將創建新的指向 OVM 的 Glance endpoint：

```
# Find Glance service id  
keystone service-list | grep glance  
# Will look similar to the following:  
| 9e539e8dee264dd9a086677427434982 | glance | image |  
  
# Add Keystone endpoint for Glance  
keystone endpoint-create \  
--service-id <GLANCE_SERVICE_ID> \  
--publicurl http://<OVM_IP>:9292 \  
--internalurl http://<OVM_IP>:9292 \  
--region <REGION_NAME> \  
--adminurl http://<OVM_IP>:9292
```

接下來刪除原有指向控制器的 Neutron endpoint：

```
# Find old Neutron endpoint id (port 9696) keystone endpoint-list # Remove old keystone endpoint for  
Neutron  
keystone endpoint-delete <NEUTRON_ENDPOINT_ID>
```



下一步創建新的指向 OVM 的 Neutron endpoint：

```
# Find Glance service id  
  
keystone service-list | grep glance  
  
# Will look similar to the following:  
| f4c4266142c742a78b330f8bafe5e49e | neutron | network |  
  
# Add Keystone endpoint for Neutron  
  
keystone endpoint-create \  
--service-id <NEUTRON_SERVICE_ID> \  
--publicurl http://<OVM_IP>:9696 \  
--internalurl http://<OVM_IP>:9696 \  
--region <REGION_NAME> \  
--adminurl http://<OVM_IP>:9696
```

Endpoint 創建完畢後，我們將用新的 Glance 主機的 Acropolis OVM 的 IP 來更新 Nova 和 Cinder 的配置。

依照下列內容編輯/etc/nova/nova.conf

```
[glance]  
  
...  
  
# Default glance hostname or IP address (string value)  
host=<OVM_IP>  
  
# Default glance port (integer value)  
port=9292  
  
...  
  
# A list of the glance api servers available to nova. Prefix  
# with https:// for ssl-based glance api servers.  
# ([hostname|ip]:port) (list value)  
api_servers=<OVM_IP>:9292
```

現在停掉 OpenStack 控制器上的 Nova 計算服務：



```
systemctl disable openstack-nova-compute.service  
systemctl stop openstack-nova-compute.service  
service openstack-nova-compute stop
```

依照下列內容編輯/etc/cinder/cinder.conf

```
# Default glance host name or IP (string value)  
glance_host=<OVM_IP>  
  
# Default glance port (integer value)  
glance_port=9292  
  
# A list of the glance API servers available to cinder  
# ([hostname|ip]:port) (list value)  
glance_api_servers=$glance_host:$glance_port
```

注釋掉 lvm 作為 cinder 後臺的選項，因為它根本不會被用到：

```
# Comment out the following lines in cinder.conf  
  
#enabled_backends=lvm  
  
#[lvm]  
  
#iscsi_helper=lioadm  
  
#volume_group=cinder-volumes  
  
#iscsi_ip_address=  
  
#volume_driver=cinder.volume.drivers.lvm.LVMVolumeDriver  
  
#volumes_dir=/var/lib/cinder/volumes  
  
#iscsi_protocol=iscsi  
  
#volume_backend_name=lvm
```

現在停掉 OpenStack 控制器上的 cinder 服務：

```
systemctl disable openstack-cinder-volume.service  
systemctl stop openstack-cinder-volume.service  
service openstack-cinder-volume stop
```

也停掉 OpenStack 上的 Glance 鏡像服務：



```
systemctl disable openstack-glance-api.service  
systemctl disable openstack-glance-registry.service  
systemctl stop openstack-glance-api.service  
systemctl stop openstack-glance-registry.service  
service openstack-glance-api stop  
service openstack-glance-registry stop
```

當所有設定檔都被更新後，我們將重啟 Nova 和 Cinder 服務使更改生效。用下列命令來重啟服務，當然也可以用腳本來完成（可供下載）。

```
# Restart Nova services  
  
service openstack-nova-api restart  
service openstack-nova-consoleauth restart  
service openstack-nova-scheduler restart  
service openstack-nova-conductor restart  
service openstack-nova-cert restart  
service openstack-nova-novncproxy restart  
  
# OR you can also use the script which can be downloaded as part of the helper tools:  
~/openstack/commands/nova-restart  
  
# Restart Cinder  
  
service openstack-cinder-api restart  
service openstack-cinder-scheduler restart  
service openstack-cinder-backup restart  
  
# OR you can also use the script which can be downloaded as part of the helper tools:  
~/openstack/commands/cinder-restart
```

15.1.1.5 Puppet

Puppet 是一個生命週期配置管理（LCM）工具，支持 devops，安全性和合規性，程式控制以及基礎架構和應用程式堆疊。

15.1.1.6 排錯與高級管理



關鍵日誌目錄

組件	關鍵日誌目錄
Keystone	/var/log/keystone/keystone.log
Horizon	/var/log/horizon/horizon.log
Nova	/var/log/nova/nova-api.log /var/log/nova/nova-scheduler.log /var/log/nova/nova-compute.log*
Swift	/var/log/swift/swift.log
Cinder	/var/log/cinder/api.log /var/log/cinder/scheduler.log /var/log/cinder/volume.log
Glance	/var/log/glance/api.log /var/log/glance/registry.log
Neutron	/var/log/neutron/server.log /var/log/neutron/dhcp-agent.log* /var/log/neutron/l3-agent.log* /var/log/neutron/metadata-agent.log* /var/log/neutron/openvswitch-agent.log*

標星號的日誌只存在於 Acropolis OVM 中。

專家提示

如果在 OpenStack Manager (Admin UI 或 CLI)中，服務被視為狀態“down”，即使服務在 OVM 中運行，也要檢查 NTP。許多服務都要求在 OpenStack 控制器和 Acropolis OVM 之間保持同步。

命令參考

裝載 Keystone 源（先於其他命令執行）

```
source keystonerc_admin
```

列出 Keystone 服務

```
keystone service-list
```



列出 Keystone endpoints

```
keystone endpoint-list
```

創建 Keystone endpoint

```
keystone endpoint-create \
--service-id=<SERVICE_ID> \
--publicurl=http://<IP:PORT> \
--internalurl=http://<IP:PORT> \
--region=<REGION_NAME> \
--adminurl=http://<IP:PORT>
```

列出 Nova 實例

```
nova list
```

顯示實例細節

```
nova show <INSTANCE_NAME>
```

列出 Nova hypervisor 主機

```
nova hypervisor-list
```

顯示 Nova hypervisor 主機細節

```
nova hypervisor-show <HOST_ID>
```

列出 Glance 鏡像

```
glance image-list
```

顯示 Glance 鏡像細節

```
glance image-show <IMAGE_ID>
```

16 後記



感謝您閱讀的 Nutanix 聖經！敬請期待更多的即將到來的更新並享受 Nutanix 平臺！