研製在 iLVM 上之遠端鏡射
Design and Implementation of an iLVM Mechanism for Remote Mirror

黃文祥　　林聖雄　　簡瑞廷　　王俊堯*　　魏守仁*
國立高雄應用科技大學電機工程系，*工業技術研究院電腦與通訊工業研究所
simon@wshlab2.ee.kuas.edu.tw

摘要

為防止大範圍災難的發生影響企業的運作，企業大多把重要的資料複製至遠端區域保存，若原有儲存設備遭受重大損壞時，企業仍可透過遠端設備維持平日正常的營運。但眾多的研究報告指出Mirror最大的挑戰是在於WAN環境上的執行能力，若傳輸延遲時間過大則將嚴重拖緩Remote Mirror的效率。本篇論文另行提出一個iLVM的Remote Backup管理機制，藉由Mirror Agent探測Local Site與數個Remote Site間讀寫傳輸速率，再依探測結果為Local Site打造出最佳的虛擬儲存裝置，Local Site的系統可藉此將資料分散至數個Remote Site進行備份保存。

關鍵詞：Remote Mirror、iLVM、Logical Storage Device

1. 簡介

近年來資料的價值對企業而言，已成為公司營運不可欠缺的因素之一，根據IDE公佈的報告表示，美國在1990至2000年中曾發生過災難的公司，約有55%立即倒閉；而剩餘的45%中，因公司營運資料遺失而兩年內倒閉的公司也約有29%，最後僅有16%的公司維持能正常運作。由此可知，為保有資料的完整性及可用性，資料備份/復原防護機制對企業而言也是相當重要的基礎建設。但在資料保護策略上，早期企業因成本與備份技術不成熟等因素考量下，公司大多在Local Site直接使用磁帶機進行備份保存。但隨著科技脈動發展，磁碟供應商陸續突破以往技術上的瓶頸限制，使得磁碟媒體在容量與價位上開始呈現反向變動趨勢，這樣的演變讓企業不再侷限僅能利用磁帶進行備份，且在備份策略上能有更多的選擇。

在Local Site所進行的備份運作，主要可避免設備故障或人為錯誤操作所導致的資料毀損，但Local Backup模式卻很難防範災難的摧殘。有鑑於此，把重要的資料分存在Remote Site儲存設備上，做更深一層的保護便有其需要。但將Local Site龐大的資料複製並傳送至Remote Site上保存，需要相當高速的網路傳輸服務才能達成此目標，所幸這幾年ISP對企業/個人用戶提供的網路服務，已從傳統DSL或專線服務漸漸汰換成FTTB(Building)、FTTH(Home)及FTTS (School)等服務。這最後一哩的網路鋪設也正式開啟低成本高頻寬的網路傳輸環境，也開始帶動異地備援的發展。

對公司而言，皆希望Remote Backup程序能避免執行周期間隔過長，導致與Local Site的資料差異過大。因此一般以磁碟為基礎的Remote Backup模式中，大多採用Mirror機制將相同的資料寫入至兩地的硬碟設備上。但傳統Mirror機制被侷限在LAN或SAN等高速率低延遲的傳輸環境上，若將Mirror機制擴展至WAN環境運行，則可因兩地地理位置距離擴大，而可保證Remote Site備援設備逃離大範圍災難機率也隨之提升。但相對Remote Mirror機制也會因增長的延遲時間而降低存取與Recovery能力。

為能有效提升Remote Mirror機制在WAN環境的執行能力，現今有許多研究與產品皆是利用Cluster RAID(Redundant Arrays of Inexpensive Disks)架構來提升Remote Mirror傳輸效能，但此架構僅支援One Site-to-One Site，且很難提供儲存資源共享與傳輸最佳化調整等功能。對此本文將提出不同於Cluster RAID的Network Storage Space Management機制，此架構無論在Distributed Network或Cluster Network環境下都能有效改善Remote Mirror執行效率。相關本文的管理架構將安排在第三章中描述，而在第二章將完整介紹Mirror與Storage Management等背景技術，此外會在此章節中簡述現今較有特色的Mirror應用研究與產品，在第四章將針對本文所提的管理架構進行一系列的實驗，最後則是本文結論及未來研究方向。

2. 背景技術

2-1 Mirror機制

“Mirror”機制顧名思義，就是讓兩個以上的硬碟如照鏡子般呈現相同的資料配置。這樣做法最大好處其可避免其一的硬碟遭受毀損時，另一者仍可持續運行避免公司資料遺失或停止作業。但由於 Mirror 機制在同一時間牽扯至兩個以上的硬碟讀寫動作，因此在執行 Mirror 機制前須規定所有成員的讀寫順序，一般較為常見的作法是將磁碟裝置區分為 Master 裝

置與 Secondary 裝置。當 System I/O Request 發送寫入訊息時，必先寫入至 Master 磁碟裝置，其次再複製 I/O Request 訊息轉送至 Secondary 磁碟進行相同寫入動作。此外 Mirror 在讀寫動作上可區分成同步(Synchronous)、非同步(Asynchronous)與半同步(Semi- synchronous)三種模式，而這三種傳輸模式其最大的差異便在於裝置寫入回覆的處理策略不同：

- **同步 Mirror－** Mirror Operator 在複製 I/O Request 請求訊息時，便轉送至所有參與成員進行相同讀寫運作，爾後所有成員完成運作後，便各自向 Mirror Operator 回應，Mirror Operator 確認所有成員皆回應才進行下次的讀寫。以確保 Mirror Pair 的裝置資料均保持一致。
- **非同步 Mirror－** 非同步 Mirror 模式與同步 Mirror 模式最大的差異，其在於 Mirror Operator 只需收到其一的成員回應便可進行下次運作。因此當 Mirror Pair 其中一者處在貧乏的傳輸環境中，非同步 Mirror 此種做法在於資料讀寫效率上遠比同步 Mirror 出色許多。
- **半同步 Mirror－** 半同步 Mirror 整體動作流程非常相似於非同步 Mirror 機制，但半同步 Mirror 將另行配置 Mirror 違約紀錄門檻值，若違約總數超越系統設定的門檻值時，Mirror Operator 將全面核對 Mirror Pair 所有成員記載資料。此作法雖也對資料可信度造成傷害，但卻能擔保傷害能降低到某種程度。

2-2 Storage Management 機制

一般儲存裝置可區分成實體儲存裝置及邏輯儲存裝置。實體儲存裝置即是一般人常見的磁碟、光碟等各類型儲存裝置；至於邏輯儲存裝置則是在實體儲存裝置與檔案系統之間擺置一個 Logical View，此 Logical View 將分別映射至數個實體儲存裝置空間區段。這種做法的好處在於可將數個實體儲存裝置合併成一台巨大的儲存裝置，或可依管理者的需求重新排列實體裝置的空間向量，且裝置讀寫運作不會與實體裝置有太大的差異。較廣為被人們使用的 Logcial View 架構有兩種，一種是 RAID 機制而另一種就是 LVM(Logical Volume Management)機制。其兩者的差異在於 LVM 架構則是將儲存裝置空間切割成等量的 Extent，再利用三層式管理架構管理所有磁碟資源，因此在 LVM 架構上可隨時新增/刪除/修改實體與邏輯儲存裝置資訊，其 LVM 的運作架構以描繪在圖 1 中。至於 RAID 架構則是則是以儲存裝置上的 Block 為基本單位。並重新排列將每個 Block Address 去建構出一個新的邏輯裝置。兩者機制相較起來，LVM 架構牽扯層面較為複雜因而效率表現較無 RAID 架構搶眼，但 LVM 三層式架構卻能比 RAID 架構更彈性的儲存空間管理。

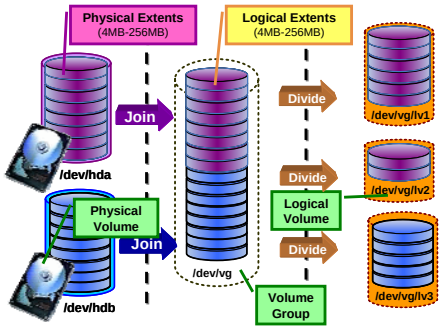

圖 1. Logical Volume Management 運作架構

2-3 相關Remote Mirror研究

為提供更有效率的 Remote Mirror 機制，美國密西根大學在 2002 年另提出 CoStore 的架構，其 CoStore 是規劃在 Cluster Network 環境，並將每台 Storage Server 視為一顆磁碟裝置以 based-RAID4 架構組織而成。當外界使用者需讀寫該 CoStore 所儲存的資料時，只需利用 NFS(Network File System) Protocol 掛載其中一台 Cluster Server 便可進行資料讀寫運作。其整個 CoStore Mirror 架構以描繪在圖 2 中。此架構最大的特點 CoStore 是利用多台 Server Parallel Transmitting 的方式來提升 Mirror 在 WAN 環境上的效能表現。此外 CoStore Mirror 在資料核對機制上，是採用核對兩地的 Parity Server 方式來確保兩地資料的準確性，藉由 Parity Code 的核對，RAID Mirror 便可知道資料是否需要啟動 Re-Synchronous。

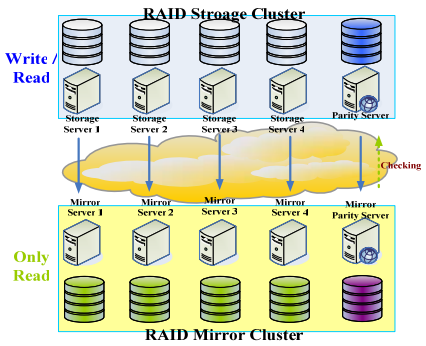

圖 2. 密西根大學的 CoStore 系統架構圖

除 CoStore 架構外，美國 Yotta Yotta 公司在 2003 也研發出 NetStorage 系統，該系統與 CoStore 一樣是同屬於 File-Level 共享儲存資源，不同的是 Yotta Yotta 採用 CFS Protocol(Cluster File System)來連貫儲存空間，且每台主機皆安裝特殊的 RAID Controller，可如圖 3 所示可直接穿越網路至遠端磁碟裝置中獲取資料。此外在 Mirror 機制上，Yotta Yotta 採用了 SAAM (Synchronous Access/Asynchronous Mirror)模式。若 Local RAID Controller 接收到主機所發送的寫入更新

時，Local RAID Controller 會預先將主機的資料寫入到 Cache 中並告知主機運作完成。此時 Local RAID Controller 會告知其他 Remote RAID Controller 變動 Page 的訊息，其他 RAID Controller 接收到此訊息後便將自身的 Cache 與磁碟中此 Page 標註成 Invalidate 並回報給 Local RAID Controller，Local RAID Controller 在接收到所有 Remote RAID Controller 的回報後在進行更新運作，如此作法可縮短 CFS Server 所進行的讀寫運作時間。

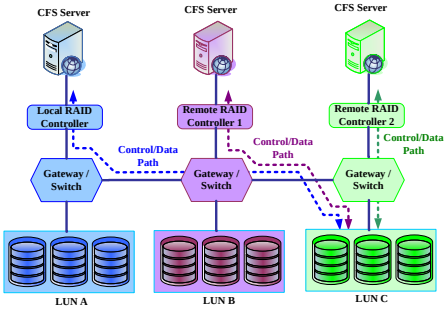

圖 3. Yotta Yotta 的 NetStorage 系統架構圖

3. iLVM 系統

密西根大學所提的CoStore架構，其特點主要是能提供One Cluster Site-to-One Cluster Site方式進行Remote Mirror機制，這樣的作法雖可以透過多台主機平行傳送程序來提升Remote Mirror效能表現，但CoStore仍需透過File-Level服務才能共享其儲存資源，且當整個Cluster RAID在行空間調整時對系統管理員而言無疑是個艱困的工程；至於Yotta Yotta所推行的NetStorage主要是憑藉著Cache的輔助才能達到如此效率，因此需要建構專用的儲存網路其成本索價不斐，這對一般中小型企業而言難以普遍使用。

3-1 系統架構

為彌補上述各機制的缺憾並能有效改善Remote Mirror Backup/Recovery的能力，本論文另提出一個iLVM(Internet Logical Volume Management)管理機制來提升Remote Mirror執行效能，iLVM機制仍是藉由傳統LVM觀念配置一個虛擬Secondary裝置，不同以往的是虛擬Secondary裝置上的LE(Logical Extents)是各Remote Mirror Site的儲存主機提供，而非以往LE是由主機自身獲得。但也因此Local Site必須要有能力知道自身所能使用的資源範圍，對此本系統將如圖4所示將在Local Site中安排一台Mirror Agent負責掌控所有儲存資源的分派，且Mirror還須定期測量紀錄Local Site與各Remote Site之間的網路狀況與平均傳輸速率，讓Mirror Agent能知道Local Site與各Remote Site的狀況以配置出最佳的虛擬Secondary裝置。因此當Local Site發送請求新的Virtual Secondary裝置時，Mirror Agent再依據自身記載的資訊去完成配置動作，爾後在回報Local Site的儲存主機最後的配置結果。

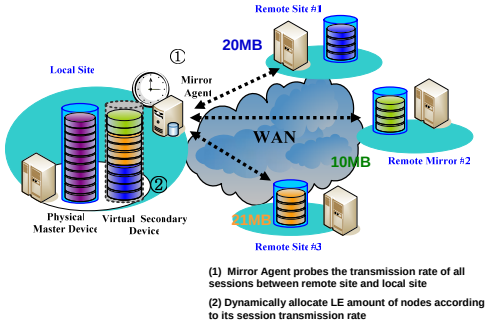

圖 4. iLVM 系統架構圖

一般公司皆希望平日投資的設備能提供更多主機彼此共用，但此模式需牽扯到multi-Site的資料維護，為此本文提出Multi-Agent的模式來滿足其需求。但在Multi-Site模式中，Mirror Agent們要顧及對稱性LE及網路狀態管理，因此在Multi-Agent環境上，系統管理者一開始需為每台Mirror Agent設定主機權重值，以便讓Mirror Agent們票選出何者為Master Mirror Agent。在Mirror Agent們再決定Master Mirror Agent的同時，Mirror Agent們將自行測量自身與其他Site之間所能達到的網路傳送速率，紀錄自身所屬的儲存主機與各Site間網路存活度。在Mirror Agent完成這些訊息收集動作後，便將這訊息一起打包送往Master Mirror Agent，再由Master Mirror Agent統一轉送給各Site的Mirror Agent，其整個動作流程可由圖5清楚得知；在LE資源配置管理方面，當Local主機發送Remote Mirror請求給Mirror Agent時，Mirror Agent須將該次配置結果轉送給所有參與的Remote Site的Mirror Agent，再由各Mirror Agent轉送給所參與的儲存主機。當Local Mirror Agent確認此次配置無任何問題發生時，Local Mirror Agent再將該配置結果送給Master Mirror Agent統一通知所有Mirror Agent進行可用LE資源更新，以確保各Mirror Agen擁有相同的資訊。

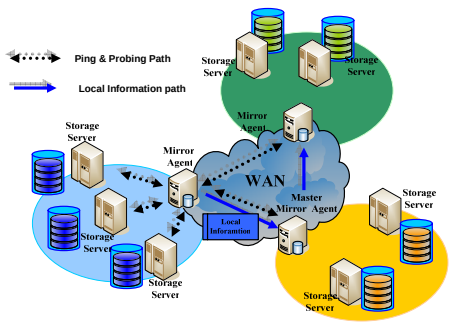

圖 5. Multi-Agent 運行架構圖

3-2 儲存空間對應(Storage Space Mapping)

由於本文的虛擬Secondary裝置其所有LE資源是從Remote Site的儲存主機所獲取，因此iLVM系統除須了解Local LE和Remote LE的對應關係外，還需明瞭如何與各Remote Site的網路建立傳輸。本文則將這些訊息皆存一個Data Node上，因此一個正常的Data Node需記載著各Remote Site的主機位址、型態、Mirror運作模式、Volume Group UUID以及LE範圍的最大和最小值，在彙整完成所有資訊後將以LE最小範圍值為索引指標，再以AVL Tree動作形式建立一個完整的Binary Search Tree。

在建構完成Binary Search Tree後，系統便可進行Remote Mirror運作。其傳統Local Mirror與Remote Mirror動作差異以圖6所示的範例說明，當傳統LVM系統在本機上執行Mirror運作時，Mirror Operator除複製I/O Request所發送的訊息外，Mirror Operator會查詢事前所建立Local Mirror Table以了解此次動作該執行在哪Mirror LE上；相對於Local Mirror運作，iLVM的Remote Mirror機制也如同Local Mirror的作法去解析Local LE該執行到哪Remote Mirror LE上，但在Mirror Operator執行LE編號置換前，Mirror Operator須先由AVL Search Tree搜尋到此段Remote Mirror LE其他額外網路傳輸資訊。在獲取這些傳輸資訊後，iLVM才得以查詢Remote Mirror表，並將I/O Request的訊息轉送至Remote Site的儲存主機執行。

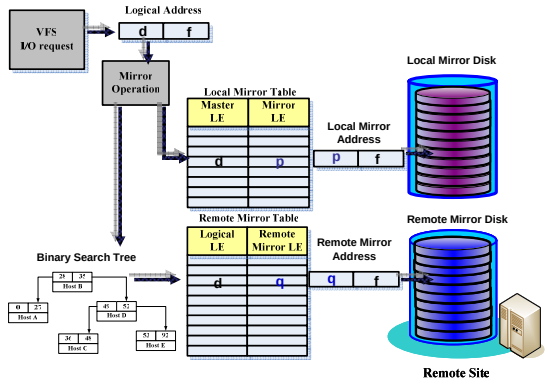

圖 6. Local Mirror 與 Remote Mirror 動作差異

3-3 空間配置演算法

對於Remote Mirror LE配置，Mirror Agent須按自身與各個Remote Site傳輸能力量身打造出最佳的Mapping Table。為此本文提出一個配置演算法可依據其所能使用的資源配置一個最佳虛擬Secondary裝置。假設Mirror Agent在接收到Local主機請求訊息時，Mirror Agent先替該Local主機尋找在各Remote Site上所能使用儲存資源主機，爾後Mirror Agent便將 i 個候選者以往的平均讀寫速率 t_i 及網路存活值 h_i 代入到公式(1)中，藉此算出每個候選者的權重值 w_i。在求得各候選者的權重值 w_i 後，Mirror Agent便為Local主機的Master儲存裝置計算LE厚度為 E_{size} 時所需的數量 N_{total}。在獲知Local主機所需的LE數量 N_{total} 及各候選者權重值 w_i 後，便將值帶入至公式(2)以求得各候選者負責的Remote Mirror LE的數量 n_i。最後Mirror Agent在依結果至各Remote Site的Mirror Agent進行申請，確認無誤後Mirror Agent便可完成Mapping Table的建立並轉送給Local主機，而Local主機也能藉此建置一虛擬 Secondary 裝置進行Remote Mirror運作。假以時日，若Local主機若因災難發生導致記載的資料慘遭毀損時，Local Site的儲存主機便可向原先建立的Secondary裝置要求進行Recovery運作。此時Remote Site的儲存主機接收Recovery請求時，各個Remote Site的儲存主機便將所屬的LE依序傳送至Local主機進行資料重建。若各Remote Site的儲存主機以單一連線進行傳送時，且Local Site頻寬 B_{local} 足以容納所有Remote Site最大單一傳輸速率，此時Local Site的儲存主機其Recovery時間 R_{time} 可由公式(3)求得。反之若Local Site端頻寬 B_{local} 不足以容納所有連線的傳送速率，則公式(3)的 t_i 總和值將由B_{local}代替。

$$w_i = t_i \times h_i \tag{1}$$

$$n_i = \frac{w_i}{\sum_{i=1}^{n} w_i} \times N_{total} \tag{2}$$

$$R_{time} = \frac{N_{total} \times E_{size}}{\sum_{i=1}^{n} t_i} \tag{3}$$

3-4 Third Mirror配置演算法

在Mirror機制中，眾多設備供商對於企業相當重要的資料將額外提供Third Mirror機制，讓企業對於資料能有更加深一層的保護。但Third Mirror機制牽扯層面過於複雜，一般在WAN環境上則不易實現，較常見的作法是由Remote Site或Local Site的自行挑選一台主機扮演Third Mirror的角色，但無論是Remote Site或Local Site所挑選出來的Third Mirror候選者，對兩者而言皆不是最佳的候選者。對此本文另提出一個演算法可在Local Site及Remote Site之間挑選出最佳Third Mirror候選人，但這演算法須限定在Multi-Agent的環境上才得以實現。以圖7的例子來說明，假設Local Site暫定Third Remote Mirror Site為Remote Mirror Site的Third Mirror候選人，此時Local Mirror Agent將查閱出 W_{RT} (Third Mirror Site與Remote Mirror Site之間的權值)、 W_{TL} (Third Mirror Site與Local Site之間的權值)，以及Local Site分配給各Site的Logical Extents數量 N_R 以及 N_T ，一起帶入公式(4)以求得Third Remote Mirror Site的權重值。

$$\varphi_{RT} = \sqrt{\left(\frac{N_R E_{size}}{W_{RT}}\right)^2 + \left(\frac{(N_R + N_T)E_{size}}{W_{TL}}\right)^2} \qquad (4)$$

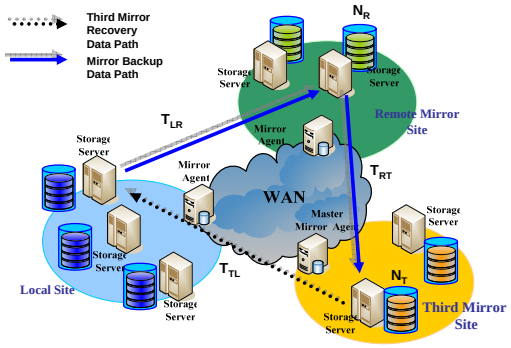

圖 7. Third Mirror 系統架構圖

由於在選擇Third Mirror候選人時，系統管理者必須注意到兩個要素。第一個要素就是Remote Mirror主機將原先的N_R個LE轉送給Third Mirror主機所需花費的時間。以及當Local和Remote Mirror主機皆故障時，Third Mirror 主機幫Local Site進行重建所需的時間。重要的是這時Third Mirror主機需傳送的資料除本身幫Local主機保存 N_T 個LE外，還有Remote Mirror轉送的 N_R 個LE。利用畢式定理的方法即可算出該路線時間斜率值 φ_{RT} 。在Mirror Agent整理出各路線的斜率值後，Mirror即可排列出一個如圖8所示的路線行列式，爾後Mirror Agent便依各Remote Mirror 5主機 N_R 量多寡順序挑選最適合的候選人，而 φ_{RT} 權值越小候選者條件越佳，但注意的是每Third Mirror候選者皆只能被挑選一次

Storage Server		Third Mirror Server S_1	S_2	S_3	S_4	S_5	S_6
	N_i	20	30	60	10	70	50
Remote Server S_1	20	0	φ_{12}	φ_{13}	φ_{14}	φ_{15}	φ_{16}
S_2	30	φ_{21}	0	φ_{23}	φ_{24}	φ_{25}	φ_{26}
S_3	60	φ_{31}	φ_{32}	0	φ_{34}	φ_{35}	φ_{36}
S_4	10	φ_{41}	φ_{42}	φ_{43}	0	φ_{45}	φ_{46}
S_5	70	φ_{51}	φ_{52}	φ_{53}	φ_{54}	0	φ_{56}
S_6	50	φ_{61}	φ_{62}	φ_{63}	φ_{64}	φ_{65}	0

圖 8. Third Mirror 運算行列式

4. iLVM測試實驗

4-1 實驗環境

為證明本文所提架構論點之可行性，在此將利用建構一個實驗網路來模擬佐證，其整個實驗網路架構以描繪在圖 9 中。在此實驗中將安排三台電腦充當 Remote Site 的儲存主機，這些主機將藉由 Intel iSCSI 透過乙太網路去提供 LE 資源。而在 Local Site 環境中則安排一台電腦扮演儲存主機及 Mirror Agent 的角色，在此台電腦除需安裝 Intel iSCSI 軟體外，還則需額外安裝 LVM 1.0.8 以及 Software RAID 套件軟體來建構一顆 Virtual Secondary 裝置。為了解這個架構所建構的 Virtual Secondary 裝置彼此間的效能差異，而在 Local 主機中安裝 tiobench 來測量一般讀寫運作的效能和延遲時間，而這些主機皆在 Linux RedHat 9.0 作業系統，Kernel 版本為 2.4.20 的環境下。此外此實驗網路需要另一台電腦負責扮演 Router 去模擬各 Site 間不同的傳輸速率，因此在該台 Router 電腦中將另行安裝 CBQ(Class Based Queuing)來達成此目標，而此 Router 電腦則是運行在作業系統 Linux Debian 3.0 其 Kernel 版本為 2.4.21 的環境下。

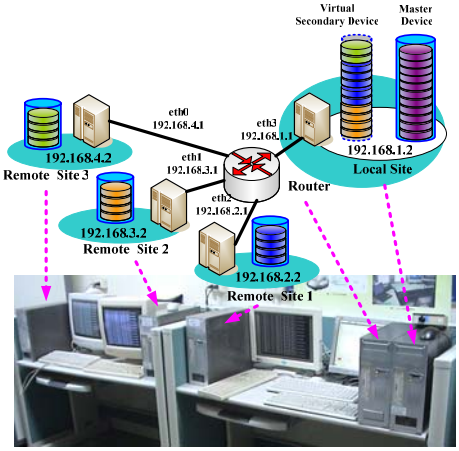

圖 9. 實驗網路環境

4-2 對稱/非對稱式傳輸環境實驗

在此實驗中，本文分別模擬兩種情結，一種情節為對稱式環境，其 Router 對各主機不限制傳輸速率，皆以 100MB/s 的環境下進行實驗；另一個情節則為非對稱式環境，限制各 Remote Site 的主機速率分別為 5MB/s、3MB/s、2MB/s。在這兩次實驗中，在表一中只列出當 Block Size 為 8192byte 的循序讀寫的測試數據為代表。在對稱式實驗中，可清楚得知 RAID-liner 模式無論在速率或延遲時間上皆能略勝 iLVM，但在非對稱的環境能量身打造 Secondary 裝置的 iLVM 將優於 RAID 機制。

表 1. 效能測試結果

		RAID		iLVM	
		Read	Write	Reads	Write
Symmetrical environment	MB/s	30.59	5.73	29.66	5.62
	ms	19.89	66.24	20.56	67.49
Asymmetrical environment	MB/s	1.04	0.48	1.62	0.68
	ms	153.3	149.91	96.635	93.657

4-3 Recovery 效率的比較

一般進行 Mirror 機制時，系統管理者除了關心 Secondary 裝置的讀寫效率外，還特別留意當 Local 主機遭受災難毀損時，各 Remote Site 再進行 Recovery 時所需花費的時間。在此實驗中先行將 Local 主機的資料全數刪除，再由各 Remote Site 的主機建構出的 Secondary 提供裝置 Recovery 複製。由於這實驗牽扯到 Block-Level 的複製動作，因此 RAID 必須如圖 10 所示須依 Block 的排列至各主機進行複製；而 iLVM 機制則可依之前的 Mapping Table 的配置，讓 Remote Site 的主機各自進行 LE 的複製，因此在 Recovery 運作時，iLVM 將明顯優異於 RAID 機制，關於 iLVM 在 Recovery 階段的效能表現以描繪在圖 11 中。

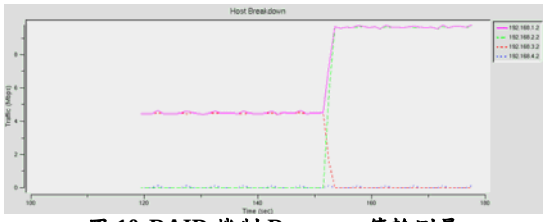

圖 10. RAID 機制 Recovery 傳輸測量

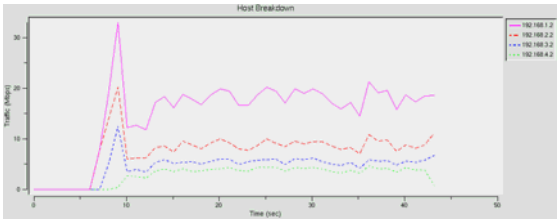

圖 11. RAID 機制 Recovery 傳輸測量

5. 結論

本論文所提的 iLVM 機制，主要是應用 Sintinat 研製的 LVM 管理架構來達成在 WAN 環境上 Remote Mirror 的資源管理，而此 iLVM 機制能依據各 Remote Site 與 Local Site 的傳輸能力及網路存活度，調配各 Remote Site 的主機所需負責的 Remote Mirror 空間的比例，為 Local Site 量身打造出最佳的 Virtual Secondary 裝置，且能讓各 Site 間彼此共享各自的儲存空間資源。此外本論文也提出適用在 iLVM 上的 Third Mirror 配置演算法，來加強 iLVM 機制在 WAN 環境上的能力。本文所提的 iLVM 機制與密西根大學所提的 CoStore 機制、Yotta Yotta 所提的 NetStorage 機制相比，其最大的差異點在於 iLVM 機制皆能適用在 LAN 或 WAN 的環境上，且也能提供 One Site-to-One Site 的模式，乃至於 Many Site-to-Many Site 下也皆能適用。為證明本論文所提的 iLVM 機制優異於其他利用 RAID 達成的 Mirror 機制，本論文也成功研製出一個 iLVM 雛型系統進行一系列的效能測試實驗，而在這些實驗所得出的結果也佐證出本論文所提的系統能真正有效提升 Remote Mirror 的能力。

6. 致謝

感謝工業技術研究院電腦與通訊工業研究所計畫支援。計畫名稱:"研究 IP 儲存應用軟體 mirror 和 Snapshot 的結構和方法"。計畫編號：T1-94026-5。計劃內容：Mirror 和 Snapshot 是產生即時儲存備份(Backup)的兩種重要不同技術。希望藉由此計畫了解儲存應用軟體對於儲存系統之影響。預期效益：縮短研發儲存系統所需應用測試程式軟體的時間，提供儲存軟體系統所需之應用測試軟體。

參考文獻

[1] Kai Hwang, Rai Jin and S.C. Ho, "Orthogonal Striping and Mirroring in Distributed RAID for I/O-Centric Cluster Computing", 2002 IEEE

[2] Yotta Yotta, High Performance NetStorage, "Data Localization and High Performance of Clustered File Systems Spanning Long Distances" October 2003 White Paper

[3] Yotta Yotta, High Performance NetStorage, "A New Paradigm for Information and Storage Sharing", October 2003

[4] Yong Chen, Lionel M. Ni, Chengzhong Xu*, Mingyao Yang†, Jay F. Kusler‡, Pei Zheng, "CoStore: A Reliable and Highly Available Storage System Using Clusters", Proceedings of the 16th Annual International Symposium on High Performance Computing Systems and Applications (HPCS.02)

[5] Marc Farley "Building Storage Network ", Second Edition, SNIA

[6] T. Yazaki, T. Kanetake, S. Akahane, Y. Sakata, K. Sugai, H. Yano, "High-Speed IPv6 Router/Switch Architecture", Proceedings of the 2004 International Symposium on Applications and the Internet Workshops (SAINTW'04)

[7] Sistina Software and RedHat, "Linux LVM HOWTO" 2004

[8] INT'L ED. "Operating System Concepts",Six Edition, WILEY

[9] UDI Manber "Introduction To Algorithm .A Creative Approach", Addison Wesley