使用前溯與多對一管理方法的遠端快照備份技術
A Novel Technique for Remote Snapshot Using Rollforward and Multiple-to-One Snapshot Management

林慶鴻 黃文祥 蔡宗霖* 蘇金沛* 魏守仁*
Ching-Hung Lin, Wen-Shyang Hwang, Tsung-Lin Tsai*, Chin-Pei Su*, and Shou-Jen Wey*

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
Department of Electrical Engineering, National Kaohsiung University of Applied Sciences
***Computer & Communications Research Lab., Industrial Technology Research Institute**
E-mail: billlin@wshlab2.ee.kuas.edu.tw

摘要

現今高度電腦化作業的企業中，資料為最重要的資產之一，因此資料的備份十分重要；快照(Snapshot)屬於線上備份技術，具有快速、免中斷服務與所需儲存空間小的優勢，成為現在資料備份的主流技術之一，除了考慮資料的備份技術外，還需考慮資料的存放地點，遠端備份(Remote Backup)提供最佳的安全性，有效避免地域性的災難；此兩項技術結合起來便成為遠端快照(Remote Snapshot)，但是由於傳統的 Remote Snapshot 的效能會受到網路的延遲時間的影響，本文即針對此課題做研究，提出一個改善的機制將網路延遲的影響減至最小，此外還提供了改良的回溯功能(Rollback)，與往前回溯(Rollforward)的功能；在 Snapshot 管理的方面，也提供了高管理性的架構，達到 Multiple-to-One Snapshot Management，可降低管理成本的效果。

關鍵詞：備份、快照、回溯、前溯

Abstract

Nowadays, data is one of the most invaluable assets in enterprises. Therefore, data maintenance is extremely important. Snapshot is a primary online backup technique with the advantages of interrupt-free, backup-fast, and space-few. Besides, remote backup became an important issue after 911 attack; it provides the highest safety to avoid regional disaster. However, the throughput of traditional remote snapshot is decreased for the increasing communication time. This paper proposes a novel technique to minimize the influence of communication time, and provide rollback, and rollforward capabilities. It also proposes improved mechanism of snapshot management: "Multiple-to-One Snapshot Management (Multiple host's snapshots store in a single snapshot volume)" to reduce the cost of management.

Keywords: backup, Snapshot, Rollback, and Rollforward.

1. 簡介

美國勞工局統計報告指出，公司如發生重大的資料損失後，有 93%的公司會在五年內結束營運[1]，因此可知資料的保護已成為企業裡不可或缺的重要任務，因此事先進行風險管理來預估潛在災難、風險分析或災難回復計畫(Disaster Recovery Plan，DRP) 是必要的，有效的選擇備份技術與事先備份資料可以有助於降低災害所帶來的損失與在最短的時間內復原。

備份機制可以分為兩類，離線備份(Offline Backup)的好處是不用考慮資料不一致的問題，備份時資料只供備份程式存取，備份時間(Backup Window)隨著資料量變多而增加，會讓 Data Loss Window 變大[3]，缺點為需要中斷服務，線上備份(Online Backup)的特色在於不需中斷服務，但會造成伺服器在服務效能上些微的下降、備份時間稍稍的拉長與備份機制上需要注意資料的一致性，企業在長時間的離線備份時損失甚鉅，因此在現今企業中，大多採用線上備份的機制。

備份技術可分為本地備份(Local Backup)與遠端備份(Remote Backup)，前者好處在於存取方便快速，然而卻無法對美國 911 般地域性災害發生保護資料的作用，遠端備份則無此之虞，且隨著網際網路頻寬的增加與租用線路所花經費的下降，讓利用網路來做遠端備份已成為趨勢。

備份模式可分為 Full Backup、Incremental Backup 與 Differential Backup，Full Backup 安全性最高，但相對需要的空間與原始資料一樣；Incremental Backup 的特點在於僅需備份與上次備份相比有改變的資料，空間的使用較少，也縮短 Backup Window，復原時需要 Last Full Backup 與其之後每一次的 Incremental Backup；Differential Backup 與 Incremental Backup 相比，則為需要備份與最後一次的 Full Backup 相比有改變的資料，因此所需的空間與備份時間較多，但是其復原較簡單，僅需 Last Full Backup 與最後一次的 Differential Backup 就可達到 Complete Recovery，Recovery Window 也較 Incremental Backup 短。

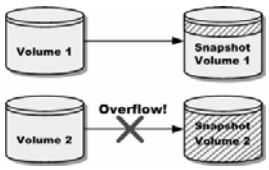
僅複製資料並無法抵抗 Data Corruption，要對抗此問題需使資料有依時間產生版本的效果，Snapshot 即可達到這項要求，SNIA 對於 Snapshot 的定義為：可以完整的呈現某個時間點上一致的可用資料的集合[4]。Snapshot 的好處為不需中斷服務、Backup Window 短(可有效的縮短 Data Loss Window)與可抵抗 Data Corruption；實作 Snapshot 的方式中，以 Copy-on-Write (CoW)為主流，因為具有良好的 Flexibility、Scalability，又可實作在 Block Level，有較高的效率；在遠端備份的情形下使用 CoW 實作的 Snapshot，由於必須額外加上兩倍的 Communication Time，造成效能的下降，本文即針對此課題，提出機制改善 Block Level 下 CoW 的 Remote Snapshot 之效能；並且加入改良的 Rollback 與 Rollforward(Rollback Forward)，在管理上採用 Multiple-to-One Snapshot Management。

以下章節依序為：第二章，介紹相關背景技術與 LSI 的 Snapshot；第三章詳述本文所提出 Remote Snapshot 機制的架構、其流程與應用；第四章，提出圖例推演與分析；第五章，為結論與未來展望。

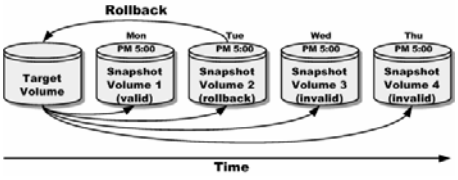
2. 背景技術

2.1 Traditional Snapshot

傳統的 Snapshot 以 LVM[5]為典型，提供 Rollback，但是所有回溯的時間點(包含此點)以後的 Snapshot 資料將會全部被刪除，如圖二所示，因此誤操作將會造成不可挽回的傷害，且 Target Volume 與 Snapshot Volume 為一對一的關係，Snapshot Volume 的空間配置有可能造成如圖一，磁碟空間利用率過低進而造成浪費，或是分配空間過小無法容納所有 Snapshot 的資料，分別調整既耗時又費工，造成管理上的不方便。



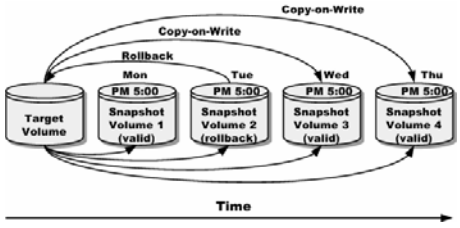
圖一、目標 LV 與 Snapshot Volume 為一對一映射



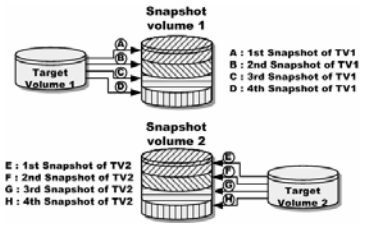
圖二、傳統 Snapshot Rollback 狀態圖

2.2 LSI's Snapshot

LSI Logic Corp.提出了改良的 Snapshot 技術，可以支援 Rollforward[6]，進而縮短 Data Loss Window 來減少損失，並可避免管理者的人為誤操作所帶來的意外；藉由保留相對於欲 Rollback 的時間點之後的 Snapshot 資料，達成可 Rollforward 的 Snapshot technique；以 Rollback 到週二下午五點的例子來說，圖三中在 LSI 所提出的機制裡，要 Rollback 之際，會逐一檢查即將被還原的 block，是否需要複製到還原的時間點後的各個 Snapshot Volume，也就是進行另外一次的 CoW，以便將來需要 Rollforward 時，可以還原到正確的狀態。



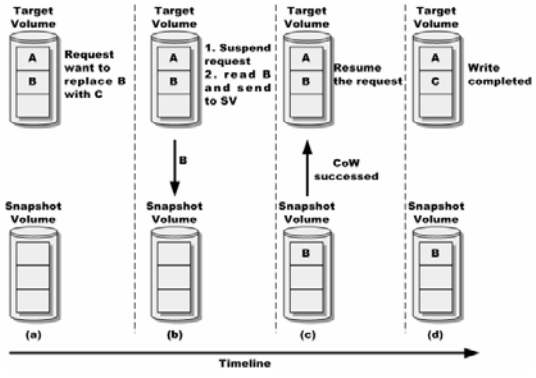
圖三、LSI 所提出 Snapshot 在 Rollback 後的狀態



圖四、單一 Snapshot Volume 存放多個快照影像

圖三中照 LSI 所提出的技術來看，以 Rollback 到 Snapshot Volume 2(週二)的例子來說，Snapshot Volume 2 的資料會被清空，以及 Rollforward 時仍要再一次的進行 Copy-on-Write，使得此一情形下的 Recovery Window 變長，另一個缺點是 CoW 的 Block 可能需要複製到多個 Snapshot Images 中，然而這兩個情形在本文所提出的架構中被改進，將於第三章的本文架構中詳述。此外 LSI 也提出了 Multiple Volumes to Single Snapshot Volume [7]，如圖四中所呈現的，支援一個 Snapshot Volume 儲存來自同一 Volume 的多個 Snapshot Images。

2.3 Traditional Remote Snapshot



圖五、傳統的 Remote Snapshot(CoW)

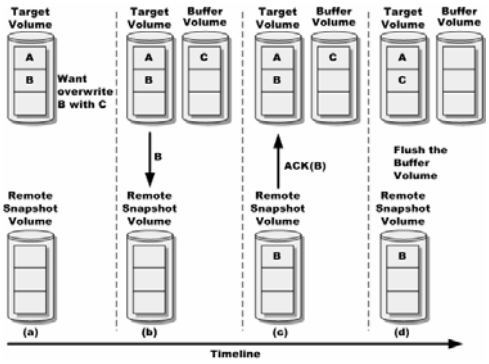
傳統 Remote Snapshot 以圖五來做解說，Target Volume 與 Snapshot Volume 分別位於 WAN 兩端的兩台不同的主機，假設 Target Volume 已有資料 A 與 B，然後針對現在的狀態進行了 Snapshot，之後發生了有一個 Request 要以 C 來取代 B，滿足了 CoW 的觸發條件，因此傳統的 Remote Snapshot 會 Suspend 發出該 Request 的 Process 並且將 B 讀取出來並且傳送到在遠端的 Snapshot Volume，並等待遠端完成寫入動作，回傳 CoW 成功後，Resume 該行程，並完成原本 C 的寫入動作，以往的 CoW 由於是在 LAN 或同一個 Storage Device 亦或是在同一電腦內的匯流排，因此 Communication Time 十分的微小，在現今講求 Remote Backup 的趨勢中，Remote Snapshot 一樣可以完成同樣的工作，但是 Communication Time 可能會使得該 Link 的 Bandwidth Utilization 過低，造成 Performance Impact 過大[8]，本文即主要針對此一課題提出新的 Remote Snapshot 機制，可以有效避免網路 Communication Time 的影響。

3. Proposed Remote Snapshot 技術

3.1 Remote Snapshot

本文架構如圖六，本地端多出一個 Buffer Volume，用來存放當 Request 觸發 CoW 時所要存放的舊資料，以前例來說，在 Target Volume 有 A、B 兩個區塊的資料後建立 Snapshot，此時接收到一個以 C 取代 B 的 Request，觸發 CoW，此時將 B 讀取並傳送給遠端的 Snapshot Volume，並"立即"將 C 重新導向寫入到 Buffer Volume，因此 Response Time 中便不會牽涉到 Communication Time。採用 Buffer Volume 的原因乃要避免寫入到原本 Target Volume 已有資料的 Block；此外，在 Buffer Volume 中的資料 Flush 到 Remote Snapshot Volume 的 Policy 則有兩點原則：

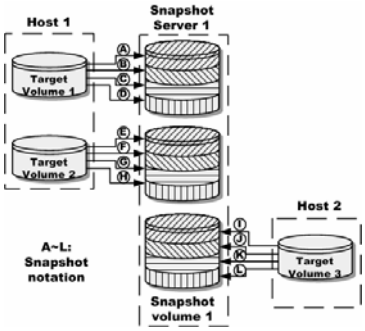
1. 遇到讀取該區塊時，一併 Flush，可以節省讀取時間。
2. 若以上的條件都未符合，則每隔一段時間就將 Buffer Volume 中的資料傳送到 Remote Snapshot Volume。



圖六、Proposed Remote Snapshot

本文所提出的 Remote Snapshot 機制可以有效縮短每個被 CoW suspend 的行程的 Response Time，並且盡量縮短、重疊與避免多餘的 I/O 動作，進而增加效能。

3.2 Many-to-One Snapshot Management



圖七、Multiple Hosts to Single Snapshot Server and Multiple Volumes to Single Snapshot Volume

為了減少多重 Snapshot 的管理成本，以達成效率與方便性，本文提出的管理機制支援不同主機同一個 Target Volume 中，不同時間點的多重 Snapshot 存放在同一個主機同一個 Snapshot Volume 中，如同圖七 Host 1 的 Target Volume 1、2 與 Host2 的四次 Snapshot 都分別存放在 Snapshot Volume 1 中，可以一次調整 Snapshot Volume 的大小，而不用如傳統 Snapshot 中需要多次調整各別多個的 Snapshot Volume。

假設在 LSI 所提出的架構中，一個 Data block 需要寫入到 n 個 Snapshot Volume，以維持 Snapshot 建立時的狀態，在本架構中，因為不管有幾個 Snapshot 要記住此 Data block 的狀態，只要一個 block 便足夠，可減少寫入次數，減少 I/O 時間，改善的百分比，同等於儲存空間節省率隨著 Snapshot 個數而上升，可以下列式子表示：

n：在 LSI 中該 Data block 須寫入到幾個 Snapshot

e：本文架構所改善的百分比(儲存空間節省率)

$$e = \frac{n-1}{n} \qquad (1)$$

在本文中，管理資訊分別存放於 RPT(Relation Pair Table)、SMT(Snapshot Management Table)、ULT(Update Location Table)；如圖七，RPT(表一)中，一個 entry 就代表了一個主機上的 Target Volume 與其 Remote Snapshot Volume 單次的映射關係，每筆 entry 都分配一個 Pair Id，用來與 SMT 產生關聯，為了降低範例的複雜度，僅使用一台主機進行多次的 Snapshot；在 SMT(表二)中的"#th Snapshot"記錄建立 Snapshot 的順序，Create Point-in-time 欄位記錄著 Snapshot 建立的時間點，I.K.為 Index Key，作用在於與 ULT 產生關聯，使得 ULT 的 entry 可以追溯到是哪一個 Target

Volume 對應到哪一個 Snapshot Volume，以及建立時間，當然 ULT 當中，要記錄該 Data block 原本在 Target Volume 中的位址，與寫入到 Snapshot Volume 時的位址；RPT、SMT 與 ULT 互相搭配，以 Pair Id 和 Index Key 產生連結的關聯，並配合所設計的演算法來達到所有的功能。

表一、RPT(Relation Pair Table)

Source Address	Target Volume	Destination Address	Snapshot Volume	Pair Id
IPv4 or IPv6	T_1	IPv4 or IPv6	S_1	P_1
IPv4 or IPv6	T_2	IPv4 or IPv6	S_2	P_2
IPv4 or IPv6	...	IPv4 or IPv6	...	...
IPv4 or IPv6	T_n	IPv4 or IPv6	S_n	P_n

表二、SMT(Snapshot Management Table)

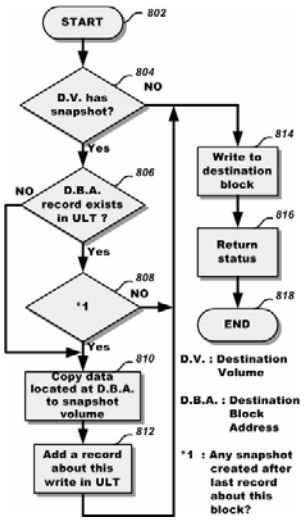
P.I.	#th Snapshot	Create Point-in-Time	I.K.
1	1	2005/03/11 17:00:00	1
1	2	2005/03/11 17:00:00	2
2	1	2005/03/10 17:00:00	3
3	1	2005/03/10 17:00:00	4

P.I.: Pair id　　I.K.: Index key

表三、ULT(Update Location Table)

I.K.	T.V. LBA	S.V. LBA
1	0x 1A-00-00-0D	0x 11-11-11-11
2	0x 2B-37-46-5F	0x 11-12-13-14
3	0x 1A-00-00-0C	0x 22-33-44-55
4	0x 9C-99-99-9D	0x 00-01-99-66

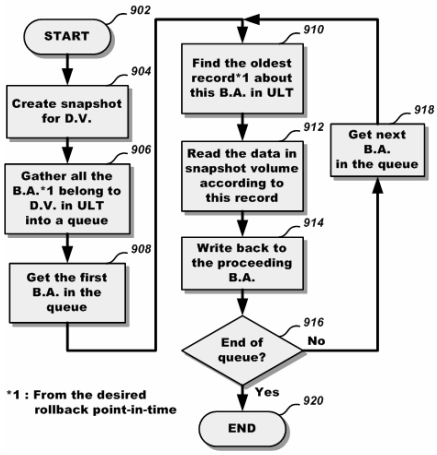
I.K.: Index key　T.V.: Target Volume
S.V.: Snapshot Volume



圖八、Write Flowchart

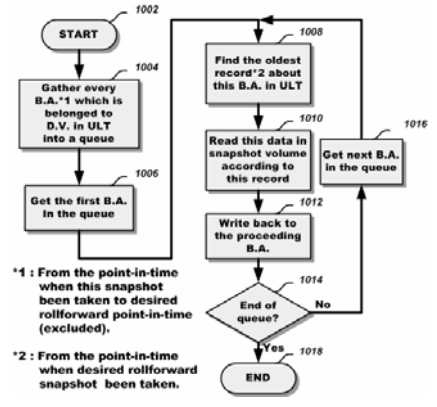
在本文的系統架構中，寫入會依據圖八的寫入流程圖來決定是否觸發 CoW，要特別注意的是步驟 **808**，如果之後並無 Snapshot 也就是不需記錄此次寫入前的狀態，便會直接寫入、回傳並離開(步驟 **814**、**816** 與 **818**)，反之即代表這次將寫入位址裡的資料應該要 CoW 到 Snapshot Volume 裡(步驟 **810**)。

在圖九 Rollback 程序裡，值得注意的是，如前例想要 Rollback 到週二下午，所收集待處理的 block address 應該是在 ULT 中屬於週二下午五點以後的記錄；與傳統 Snapshot、LSI 所提出的 Snapshot 不同之處在於：Rollback 後該 Snapshot 資料的處理，在本架構中不予刪除，可提供較高的資料可獲得性，並且有較短的復原時間，再進行其他的復原工作時，也省去了 Copy-on-Write 的步驟，有助達到 RTO(Recovery Time Objective)。



圖九、Rollback flowchart

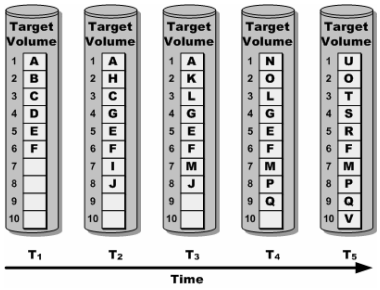
Rollforward 原則與 Rollback 一致，但哪些 block 被改寫過與存放要 Rollforward 的 Data block address 的搜尋範圍不同，以圖十的流程圖來說，以週一到週四每天下午五點都建立一次 Snapshot 的例子來說，假使已經 Rollback 到週二下午，如果要 Rollforward 到週四的話，便要到 ULT 中收集時間為週二與週三的記錄的 block address，到佇列中，然後在(步驟 **1008**)ULT 中尋找該位址在欲 Rollforward 時間點以後最舊的更新記錄，也就是要找週四以後屬於該 block 最舊的更新記錄，接下來便照著圖十我們所設計的流程執行直到結束。



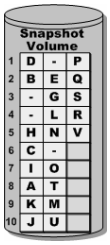
圖十、Rollforward flowchart

4. 圖例推演

此處以 LSI 的例子[6]來驗證，首先假設有一 Target Volume(含有 Primary data)，其資料變更過程如圖十一所示，分別在 T_1 到 T_4 時建立了共四次的 Snapshot，時間為 2005/03/01~2005/03/04 每天下午五點，當時間為 T_5 時，進行 Rollback 到 2005/03/02 下午五點，然後再 Rollforward 到 2005/03/04 下午五點；分成三個部分，第一部分為 T_1 到 T_5，Target Volume 部分被改寫，Snapshot Volume 與 ULT 的內容漸增，T_1 到 T_2 間，G、H、I、J 分別依序寫入位址 4 、2、.7、8，G 要寫入時會啟動 CoW，將位址 4 的 D 複製到 Snapshot Volume 中第一個位址(參考圖十二)，並在 ULT 中添加 Index Key 為 1、Target Volume LBA 為 4、Snapshot Volume LBA 為 1 的記錄(參考表六)，H、I 與 J 寫入時也啟動了 CoW，分別在 ULT 中增加了 T.V. LBA 為 2、7、8，S.V. LBA 為 2、3、4 的記錄，Snapshot Volume 則保留了 B、-(表無資料)與-，接下來 T_2 到 T_5 則以此類推。



圖十一、圖例的 Target Volume 變化圖



圖十二、圖例的 Snapshot Volume

表四、圖例的 RPT

Target Volume uuid	P.I.
aV776f-UgE8-FtF1-xN6s-Wvwb-D1Qq-7xRxw9	1
Snapshot Volume uuid	
XV1Rcr-AzZ9-srak-KTX9-Ikfk-174o-wy50A2	

表五、圖例的 SMT

P.I.	#th snapshot	Create Point-in-time	I.K.
1	1	2005/03/01 17:00:00	1
1	2	2005/03/02 17:00:00	2
1	3	2005/03/03 17:00:00	3
1	4	2005/03/04 17:00:00	4
1	5	2005/03/04 20:00:00	5

表六、圖例的 ULT

I.K.	T.V. LBA	S.V. LBA	I.K.	T.V. LBA	S.V. LBA	I.K.	T.V. LBA	S.V. LBA
1	4	1	3	8	10	5	7	19
1	2	2	3	9	11	5	1	20
1	7	3	4	5	12	5	8	21
1	8	4	4	4	13	5	9	22
2	2	5	4	3	14	5	5	23
2	3	6	4	1	15	5	4	24
2	7	7	4	10	16	5	10	25
3	1	8	5	2	17	5		
3	2	9	5	3	18	5		

接下來假設在時間為 2005/03/04 晚上八點時進行 Rollback 到 T_2 的步驟，由圖九的 Rollback 程序，先到 SMT 中加入一筆 Snapshot 的記錄，然後依步驟 **606** 收集待處理的位址到佇列中(如表七)，在每一位址還原前，都要 Copy-on-Write 到 Snapshot Volume，因為屬於 T_5 的狀態，將 O、T、M、U、P、Q、S、R、V 分別存入 Snapshot Volume 中的 17~25 位址，如此一來 Rollback 程序完成。

表七、Rollback 到 T_2 的處理佇列

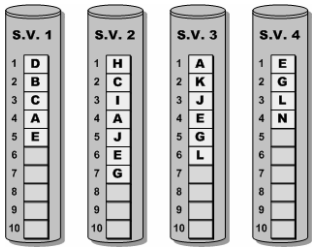
2	3	7	1	8	9	5	4	10

現在 Target Volume 屬於 T_2 的狀態，假使現在需要 Rollforward 到 T_4 的狀態，依圖十的步驟 **704** 收集所有屬於 T_2 與 T_3 紀錄的位址成為表八的佇列，然後再到 ULT 中，尋找每一位址在 T_4 以後的最舊一筆的記錄(步驟 **708**)，直接根據該筆記錄將資料讀出然後寫回 Target Volume，直到所有佇列中的位址都處理完成便完成 Rollforward 程序。

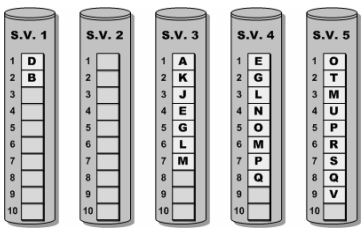
表八、Rollforward 到 T_4 的處理佇列

2	3	7	1	8	9

5. 效能比較



圖十三、LSI 架構於 T_5 時之狀態



圖十四、LSI 架構在 T_5 Rollback 到時 T_2 後的變化

，主要分為兩個類別，使用空間、R/W(Read/Write)次數，三個部分來與 LSI 的 Snapshot 技術做比較，分別為：

1. T_5 與 Rollback 到 T_2 時 Snapshot Volume 的 block 使用數。
2. T_5 Rollback 到 T_2 所需的 R/W 的次數
3. 由 T_2 Rollforward 到 T_4 所需的 R/W 的次數

表九、block 使用數比較表

Technique \ Used blocks	T_5	Rollback to T_2
LSI	22	26
Present	16	25

表十、T_5 Rollback 到 T_2 所需的 R/W 次數比較表

Technique \ R/W times	T_5 Rollback to T_2
LSI	36
Present	36

由表九，可以得知，縱使不將 T_2 的 Snapshot 資料刪除，仍然使用較少的空間，而且不將 T_2 的 Snapshot 資料刪除將會使得之後 Rollforward 時減少 R/W 的次數，達成縮短時間的效果。

表十一、T_2 Rollforward 到 T_4 的 R/W 次數比較表

Technique \ R/W times	T_2 Rollforward to T_4
LSI	22+
Present	12

表十中，LSI 將 Snapshot Volume 2 的七個 block 分別寫回 Target Volume，H、C、I、A、J、E、G 7 個 block 寫回時，皆須將原位址的資料 CoW 至 Snapshot Volume 5，由於牽涉到 CoW，所以還原一個 block 要 4 次的 R/W(2 reads and 2 writes)，加上 Q、V 各 2 次的 R/W(無還原，只做 CoW)，再加上多寫入 1 次到 Snapshot Volume 3 的 M，與多寫入到 Snapshot Volume 4 各一次的 M、P、Q，共為 (7×4)+(2×2)+1+(1+1+1)=36 次；本文所提出的方法，如表七所示 Rollback 到 T_2 共需處理 9 個 block，且每次都有 CoW，因此 R/W 次數為 9×4=36 次。

表十一中，LSI 的部份，Snapshot Volume 中有 8 個 block，其中只有 5 個還原時會牽涉到 Copy-on-Write，另 1 個會直接讀出並寫入，所以至少為 5×4+2=22 次；在本文提出的架構中，如表八所示，僅需 6 個 block 的讀出與寫入(無需將原有資料 CoW)，所以其 R/W 為 6×2=12 次。

6. 結論與未來展望

本文提出了一個以遠端快照為主的架構，並將網路的 Communication Time 的影響降至最低，對於採用 Remote Backup 有極大的幫助，本文架構的優點如下：

1. 可以有效避免 Remote Snapshot 時，網路傳輸延遲所造成的效能影響。
2. 集中管理同一 Target Volume 的 Snapshot，可以縮短調整時間，減少管理成本。
3. 較 LSI Snapshot 技術節省空間。
4. 有效減少 I/O 的時間，縮短回復時間。

本文架構現處於 Block Level，優點是不用受限於 File System 的格式，作者發現在 File System Level 仍有發展潛力，將會鑽研其可行性，並找出更新週期時間參數的最佳值；此外此系統的 Scalability 也是一個值得探討的課題。工研院電通所近年積極投入 Mirror 和 Snapshot 技術的開發與軟體的研製，未來將與工研院合作之下，針對本文所提出架構進行實作。

7. 感謝

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] Bud Stoddard, “The growing role of online server Data backup and recovery,” in Storage Networking World Online, April 7th, 2003.
[2] The Business Continuity Institute, “Business continuity management”, http://www.thebci.org/
[3] Elisabeth Horwitt, “Snapshots undo virus damage,“ in Storage Networking World Online, January 3rd, 2005.
[4] “SNIA – Storage Networking Industry Association”, http://www.snia.org
[5] Logical Volume Manager - LVM2 Resource Page, http://sources.redhat.com/lvm2/
[6] Huber, Robin, Humlicek, and Donald R., "Data timeline management using Snapshot Volumes," United States Patent No. 6,771,843, LSI Logic Corporation, August 3, 2004
[7] Donald R. Humlicek, Rodney A. DeKoning, William P. Delaney, “Managing a snapshot volume or one or more checkpoint volumes with multiple point-in-time images in a single repository," United States Patent No. 6594744, LSI Logic Corporation, July 15, 2003
[8] “Global Data Sharing Result”, http://www.yottayotta.com