使用隧道技術實現網路虛擬化

曾惠敏　　胡仁維　　劉德隆
財團法人國家實驗研究院國家高速網路與計算中心
{ n00hmt00, hujw, tlliu }@narlabs.org.tw

摘要

近年來隨著雲端運算（Cloud Computing）的蓬勃發展，虛擬化（Virtualization）技術成為熱門的研究議題，其中主機虛擬化由於可將單一實體主機幻化為多部共同運作中的虛擬主機（Virtual Machine，VM），現已有多種商業與開放原始碼的虛擬化系統（Hypervisor）解決方案，可謂於雲端產業中最為成熟的技術之一；相較於虛擬主機可直接服務提供給終端使用者，位於底層的網路資源之虛擬化的需求度較為不高，但隨著跨機房的資源整合與大型虛擬環境的要求日漸成形，網路虛擬化（Network Virtualization）技術也在不斷摸索中逐漸成形，以隧道（Tunneling）技術動態提供虛擬網路資源已成為市場主流，在本篇論文中，我們將介紹網路虛擬化演進的過程，並分析目前主流的隧道技術，同時以開放原始碼的 Open vSwitch 結合 KVM 來實作出各種虛擬主機與網路配置的案例，提供整合虛擬網路與主機資源規劃之參考。

關鍵詞：網路虛擬化、雲端運算、Open vSwitch、軟體交換器、軟體定義網路。

1. 前言

隨著雲端時代的到來，隨用隨租的運作模式已徹底改變大眾對於數位服務的觀念，例如：可以依照自己的需求（CPU 數量、記憶體大小等）來訂製臨時使用的計算主機，使用完後歸還並依使用量或時間計費，省去自行購置及維護的成本與風險[1]。在雲端服務的背後，資源虛擬化被視為重要的技術之一，透過將實體資源虛擬化的技術，服務供應者方能對使用者的需要進行客製化，並可隨時動態地配置。主機虛擬化技術可以將實體主機依客戶需求同時配置成多部虛擬主機，市面上已有多個商業及開放原始碼的解決方案可採用，目前成為相當成熟的技術，並為大部份的基礎設施即服務（Infrastructure as a Service，IaaS）業者所採用，相對而言，網路資源由於位於底層，一般使用者較無客製化的需求，因此網路虛擬化的成長遲遲並未跟上雲端技術的發展[2][3][4]。

然而因應虛擬主機服務的擴展，虛擬主機間的動態網路配置益發顯示其重要性：如同一使用者所建之虛擬主機若跨不同的實體主機，則需要在實體主機間配置虛擬網路；以及不同使用者的虛擬主機之間需要區隔不同的網路網段等，為此網路虛擬化技術漸獲重視，但存在於實體主機內的軟體交換器與原先實體的硬體交換器間的互通模式需要由主機管理員與網路管理員彼此溝通，造成管理上的複雜度，為此已有不同陣營的網通廠商各自提出 IEEE 802.1Qbg[5]及 802.1BR[6]（取代 802.1Qbh[7]）等標準，試圖將網路設定的主導權自軟體交換器歸還給外部的硬體交換器[8]，但是由於缺乏統一標準，且仍需要與虛擬主機技術間整合，因此未獲大幅度之採用。

不論軟體或硬體式的交換器，基本上還是以 802.1Q 虛擬區域網路技術（Virtual Local Area Network，VLAN）來區分不同的使用者，然而 VLAN 的識別碼僅 12 位元，僅能區隔最多 4094 個用戶，無法因應大量需求；且第二層交換所使用的擴張樹路徑搜尋法也讓可用頻寬受限，無法發揮網路最大的使用率；同時數量龐大的虛擬機器也將會造成機架頂端（Top of Rack，ToR）之交換機過量的負荷，第二層位置表格的容量因受限記憶體大小將不敷使用，因此既有的 VLAN 方案無法滿足雲端環境所需要的虛擬網路需求，有鑑於此，虛擬機器透過軟體交換器間的虛擬隧道建立連線成為快速解決的途徑，相關設定均在伺服器端即可完成，底層網路僅為傳遞封包通透使用，無需額外設定。

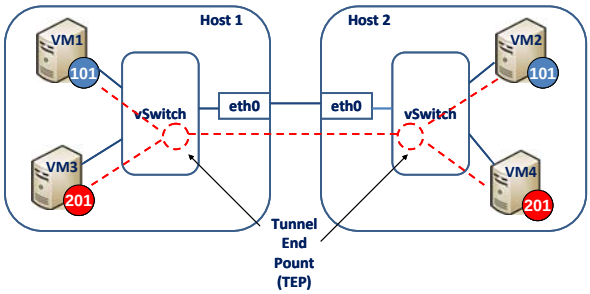

圖 1、 隧道技術示意圖

圖 1 為隧道技術的示意圖，兩個實體主機 Host1 與 Host2 內部的虛擬網路交換器透過底層網路建立起點對點的虛擬通道連線，通道兩端點通常稱為隧道終端節點（Tunnel End Point，TEP），各交換器在傳送封包到隧道之前會依照不同的使用者加註識別碼，如在圖 1 中 VM1 欲傳送封包至 VM2，Host1 內的軟體交換器會在包裝隧道封包時將使用者識別碼 101 填入，傳送到 Host2 後根據此識別碼正確地將封包送給 VM2，如此一來可有效地區隔使

用者識別碼為 101 的 VM1、VM2 與使用者識別碼為 201 的 VM3、VM4。

在本篇論文的第 2 節中，我們將綜覽目前主流的隧道技術並進行比較；第 3 節將介紹如何使用 Open vSwitch 實作出 GRE 隧道，同時進行簡單的效能實測；在第 4 節中，我們將針對雲端環境下的各種虛擬網路情境提出佈建之範例；並於第 5 節作出本篇論文的結論。

2. 隧道技術縱覽

本章節介紹目前三個主流的隧道技術，包括虛擬可擴展式區域網路（Virtual eXtensible Local Area Network，VXLAN）[9]、透過一般路由封裝實現網路虛擬化（Network Virtualization using Generic Routing Encapsulation，NVGRE）[10]、以及無狀態式傳輸層隧道（Stateless Transport Tunneling，STT）[11]，三者源自不同的公司，各自使用選擇不同的方式進行隧道封裝，目前均為 IETF 的 Internet Draft，以下各別敘述其細節，並整理三種技術間的比較。

2.1 VXLAN－虛擬可擴展式區域網路

VXLAN 為 VMWare 與 Cisco 等公司於 2011 年 8 月所草擬，目前最新版本為第四版，為三者當中最早提出的版本，點對點隧道的兩端點稱為 VXLAN 隧道終端節點（VXLAN Tunnel End Point，VTEP），並使用 UDP 進行隧道封裝，如圖 2 所示。

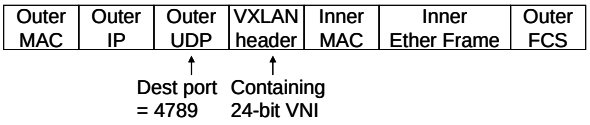

圖 2、 VXLAN 封包格式

被封裝的第二層封包由 VXLAN 表頭所包覆，在 VXLAN 表頭裡有個長度為 24 位元的 VXLAN 網路識別號碼（VXLAN Network ID，VNI），用來區隔使用該隧道的使用者，因此理論上可支援 2^{24} 超過一千六百萬個不同虛擬網路，遠大於 VLAN 4094 個的限制。整個 VXLAN 封包再由外層的 IP/UDP 封包透過第三層實體網路在兩個 VTEP 間傳遞，在 UDP 的表頭裡的目的地埠號已向 IANA 申請了 4789 作為辨識 VXLAN 的依據。而為了在多路徑的網路環境下達到負載平衡，即等價多路徑路由（Equal Cost Multi-Path，ECMP），建議 VTEP 在設定 UDP 的來源端埠號時參考內部被封裝的封包表頭的雜湊值，可依此區別不同資料流（flow），確保屬於相同資料流的封包在第三層傳輸的路徑相同，並讓不同資料流的封包分散至等價的路徑上。

在廣播（broadcast）與群播（multicast）的處理上，VXLAN 要求底層的第三層網路啟用 IP 群播功能，在 VTEP 上透過 VNI 與 IP 群播群組的對照表格處理各虛擬網路的廣播與群播封包，然而現今網際網路骨幹上大多並未開啟 IP 群播功能，這點可能會讓 VXLAN 在大規模佈建時受到限制。

2.2 NVGRE－透過一般路由封裝實現網路虛擬化

NVGRE 是由 Microsoft 為首的公司於 2011 年 9 月所提出，僅略晚 VXLAN 不到一個月，目前已修訂到第三版，不同於 VXLAN 使用 UDP 封包格式，NVGRE 採用現有的一般路由封裝（Generic Routing Encapsulation，GRE）之格式，隧道兩端點稱為網路虛擬化端點（Network Virtualization Edge，NVE）。

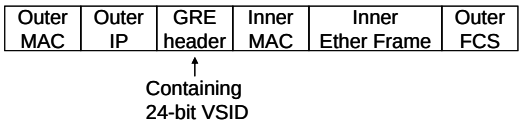

圖 3、 NVGRE 封包格式

NVGRE 封包格式如圖 3 所示，由 GRE 表頭來包覆被封裝的第二層封包，但是與 VXLAN 不同，內部的第二層封包並不支援 802.1Q VLAN 標籤功能，此部份可能會讓虛擬網路的上層應用受到限制，在 GRE 表頭裡定義的 32 位元 Key 欄位在此被拿來作為 24 位元的虛擬子網路識別碼（Virtual Subnet ID，VSID），以及 8 位元的網路流識別碼（FlowID），VSID 的用途與 VXLAN 裡的 VNI 相同，可區別超過一千六百萬個不同虛擬網路；FlowID 則是用來識別不同的資料流，然而現有的等價多路徑路由機制僅根據 TCP/IP 表頭來辨識資料流，並不支援 GRE 表頭，因此在底層設備並未支援 FlowID 的檢視之前，NVGRE 建議使用多個 NVE 實體 IP 位置來對應不同的資料流，在負載平衡上的處理仍然不夠細緻。

在廣播和群播的處理上，NVGRE 可使用 IP 群播的機制，若底層網路不支援 IP 群播，亦可用多重單點傳送來達成相同功能；同時考量隧道封包機制會將增加整體封包長度，若長度超過底層網路的最大傳輸單元（Maximum Transmission Unit，MTU）時封包會被強制分段（fragmentation）至零碎的封包，這些封包於隧道的另一端接收後再進行重組工作，將會增加隧道端點的負荷並降低網路傳輸效率，NVGRE 在內部封裝為 IP 封包時會辨識網路路徑 MTU（Path MTU），若大於封裝後的長度則會強迫封包維持原狀以避免此分段的情境，以上兩點均讓 NVGRE 適於在現有網際網路上傳遞。

2.3 STT－無狀態式傳輸層隧道

STT 是由新創公司 Nicira 於 2012 年 2 月所單獨提出，並被該公司申請專利，目前更新為第三

版，同時由於 Nicira 已於 2012 年 7 月被 VMWare 所併購，目前本 Internet Draft 也改為由 VMWare 所提案。STT 主要是要解決 2.2 節裡所述封包分段的問題，雖然 NVGRE 已盡量嘗試避免，但封包仍然會有被分段的可能，為此，STT 利用現今網路卡對 TCP 分段卸載（TCP Segmentation Offload，TSO）的支援功能，將欲封裝的封包包裝於類 TCP（TCP-like）的表頭之內，如此一來封包的分段及重組的工作均交由硬體網路卡處理，不會影響到隧道端點的運作效能，已有國外的技術網站[12]針對使 Open vSwitch 進行測試比較，發現 STT 完全不影響網路傳輸吞吐量與 CPU 效能，而相同情況下使用 GRE 封裝時傳輸吞吐量的表現僅 1/4 且 CPU 負載亦飆高到 97%，足見 STT 在技術上的優越性。

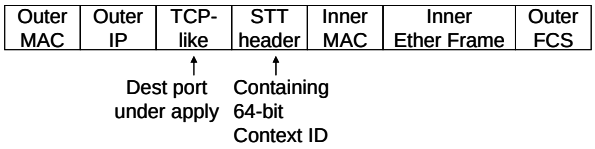

圖 4、 STT 封包格式

STT 封包格式如圖 4 所示，在 STT 表頭內保留了 64 位元的關聯識別碼（Context ID），因此可辨識的虛擬網路數量更為可觀，而在 TCP-like 表頭的目的地埠號正向 IANA 申請固定號碼中，以作為辨識 STT 的依據，來源埠號則是與 VXLAN 裡 UDP 來源埠號一樣，可用來作為等價多路徑路由的依據，在廣播與群播的處理上則是與 NVGRE 相同，底層的網路不必然一定要支援 IP 群播能力。

STT 的獨特設計讓 Nicira 公司大受矚目，並視為市場上其他雲端軟體公司的勁敵，因此在被 VMWare 收購時亦造成熱門之話題，目前 VMWare 同時擁有 XVLAN 與 STT 兩個相同的技術，未來是否會將兩者互相整合也受到市場的關注。

表 1、 三種隧道技術比較

	VXLAN	NVGRE	STT
Encapsulation	UDP	GRE	TCP-like
# of Virtual Networks	2^{24}	2^{24}	2^{64}
VLAN-tag allowed in inner Packet	Yes	No	Yes
IP Multicast Required	Yes	No	No
ECMP Granularity	Low	High	Low
Performance	Poor	Good only when inner packet is IP and path MTU allows	Good

綜合以上之討論，我們將三種隧道技術的比較整理如表 1，比較的項目分別為使用的封裝技術、支援虛擬網路之數量、內部封包是否可使用 VLAN 標籤、底層網路是否需要開啟 IP 群播功能、等價多路徑路由之細緻度、以及實際運作效能等。

3. 以 Open vSwitch 實作 GRE 隧道

Open vSwitch[13]為美國史丹福大學所開發的開放原始碼虛擬交換器軟體，可取代傳統 linux 內建橋接器（Bridge）模組，以取得更好的交換效能，同時 Open vSwitch 亦支援以 VLAN 標籤區隔不同用戶，以及 OpenFlow 這個重要的軟體定義網路（Software Defined Network，SDN）協定，因此頗受各界重視。

關於在第 2 節中所提到的三種隧道封裝格式，GRE 在 Open vSwitch 開發初期就已支援；而在今年 5 月最新的 1.10.0 版本裡亦正式支援了 VXLAN 格式；另外由於 Nicira 公司同為史丹福大學團隊所創立，STT 商業版本的實作亦選擇 Open vSwitch 為搭配之交換器，同時因專利擁有權之故，其他公司無法在自有的交換器上使用 STT 技術，因此成為目前唯一擁有三種封裝技術的解決方案。本節將以 GRE 封裝為例介紹如何在兩台 Open vSwitch 間組態專屬隧道，這邊要注意的是 Open vSwitch 僅分別支援 UDP 與 GRE 的封裝格式，並未完整實作出 VXLAN 與 NVGRE 的協定，因此我們僅能建立點對點隧道，需額外撰寫控制模組以使用 VNI 與 VSID 等欄位及其他相關功能，這部份可透過軟體定義網路 SDN 或自行設計控制層界面，而控制層軟體是否能夠統一標準也將是隧道技術未來發展的重要關鍵。

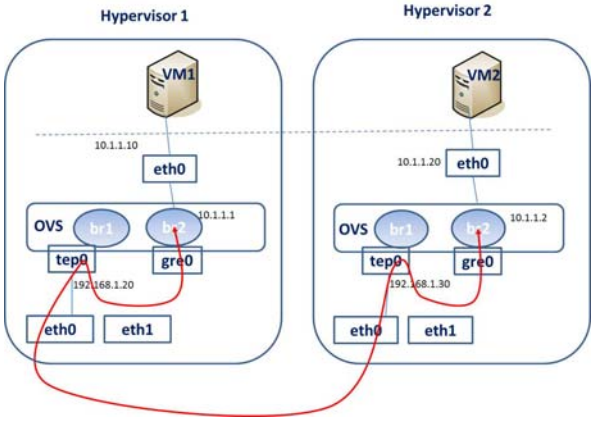

圖 5、點對點 GRE 隧道測試環境

圖 5 為實作的架構圖，在我們的測試環境中，包含兩台實體主機分別為 Hypervisor1 和 Hypervisor2，使用 KVM 配置虛擬主機，其中每台主機下各執行一個虛擬機器分別為 VM1 和 VM2，我們將測試透過 Open vSwitch 使用 GRE Tunneling 的技術來實作虛擬主機的網路配置以達到 VM1 和 VM2 互通目的。然而使用 libvirt 虛擬化管理介面建立的虛擬機器，預設網路介面會被自動地分配一個 vnetx 的網路名稱，經由 bridge(virbr0)連接出去，為了將其整合於 Open vSwitch 中管理，並指定須經由

OVS bridge(br2)連接出去。故須於/etc/libvirt/qemu 下產生之相關的 XML 描述檔修改設定，如圖 6，待修改後重新啟動虛擬主機時會載入新的 XML 描述檔，即完成該虛擬主機之網路介面整合於 Open vSwitch 中，如圖 7 所示。接著我們可依相關需求使用 Open vSwitch 指令進行相關 GRE Tunneling 的設定。

```
<interface type='bridge'>
  <mac address='52:54:00:84:13:17'/>
  <source bridge='br2'/>
  <virtualport type='openvswitch'>
    <parameters interfaceid='70b99bbf-edaf-b37c-c913-7be61997f61a'/>
  </virtualport>
```

圖 6、 Open vSwitch 指令

```
Bridge "br2"
    Port "gre2"
        Interface "gre2"
            type: gre
            options: {remote_ip="192.168.1.40"}
    Port "vnet0"
        Interface "vnet0"
```

圖 7、 整合於 Open vSwitch 之介面

建立 GRE Tunneling 有三個基本的步驟：

1. 新增一個獨立的 Bridge。
2. 在每一台 Hypervisor 上建立 GRE Tunneling 使用之 endpoint。
3. 建立欲連接使用之 GRE Tunneling。

這裡，我們需要在兩台實體主機上分別新增 OVS bridge(br1)和設定其使用之介面為 eth0，並且在該 OVS bridge(br1)上指定 GRE Tunneling 連接使用之 endpoint(tep0)，另外新增一個獨立的 OVS bridge(br2)，用來建立 GRE Tunneling (gre0)，圖 8 為所設定的 Open vSwitch 指令。

```
# sudo ovs-vsctl add-br br1
# sudo ovs-vsctl add-port br1 eth0
# sudo ovs-vsctl add-br br2
# sudo ifconfig br1 211.73.95.33 netmask 255.255.255.224
# sudo ifconfig br2 10.1.1.1 netmask 255.255.255.0
# sudo ovs-vsctl add-port br1 tep0 -- set interface tep0 type=internal
# sudo ifconfig tep0 192.168.1.20 netmask 255.255.255.0
# sudo ovs-vsctl add-port br2 gre0 -- set interface gre0 type=gre
options:remote_ip=192.168.1.30
```

圖 8、 Open vSwitch 指令

上列指令設定完成後，我們可以測試由 VM1 執行 Ping 指令到 VM2 驗證透過 GRE Tunneling 得到網路互通。實際上就像經由兩個 bridge 直接將網路互相連接，如圖 9。

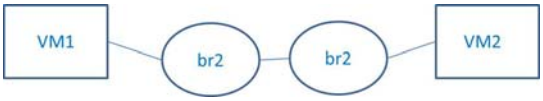

圖 9、 點對點虛擬連接拓樸圖

在 2.3 節裡我們提到[12]已有針對 STT 與 GRE 的效能進行測試，GRE 效能不彰的原因主要是在於封包分段所需的軟體處理上，使得網路吞吐量降低，也使 CPU 使用率飆高，雖然在 NVGRE 標準中為了避免此狀況有進行路徑最大傳輸單元的偵測，但並未詳述實作之細節，因此我們手動將兩實體主機和中間的交換器之最大傳輸單元依巨型封包的定義（Jumbo Frame）[14]設定為 9000，以確保來自 VM 的封包不會被分段，採用 iperf 軟體使用 TCP 單線傳輸，針對有無使用 GRE 隧道及最大傳輸單元為 1500 或 9000 共四種狀況進行測試，結果如表 2 所示。

由表 2 中我們可以看到在一般未經隧道技術及最大傳輸單元為 1500 的情況時，在 1Gbps 的網路環境下可到達 940 Mbps 的效能；當最大傳輸單元調高為 9000 時，可以看到效能隨著單一 TCP 封包傳遞的內容增加而增進到 990 Mbps，但頻繁的傳輸也讓 CPU 使用率由原來的 6%增加到 15%；若使用 GRE 隧道，在原先最大傳輸單元為 1500 的狀況下可以發現網路效能因封包分段的處理而降低到 710 Mbps，且 CPU 使用率也因軟體的隧道封裝增加到 22%；在設定最大傳輸單元為 9000 後，因為解決了封包分段的問題，效能可恢復到 978 Mbps，但 CPU 的使用率也因同時處理隧道及較高的傳輸速度而飆升至 30%。

藉由以上的實測，我們可知調整最大傳輸單元可以有效地解決隧道技術中傳輸效能不彰的問題，但是可能會加重 CPU 的負荷；同時在實務環境裡，目前除了研究網路骨幹外，一般的網際網路環境並未全面採用巨型封包的最大傳輸單元，所以此方式僅限於資料中心或是公司網路內部，若要建立跨網域的隧道，中間只要有一個網路設備沒有調整最大傳輸單元的話，所有長度超過大小的封包都會被丟棄導致無法通訊，所以斧底抽薪的解決方法仍待未來新一代硬體網路卡提供隧道封包的卸載功能，讓針對隧道的軟體處理均交由底層網卡負責，方能讓隧道技術的建置成熟。

表 2、 GRE 隧道效能測試

	TCP Throughput	CPU Utilization
w/o Tunnel MTU=1500	940 Mbits/sec	6%
w/o Tunnel MTU=9000	990 Mbits/sec	15%
GRE Tunnel MTU=1500	710 Mbits/sec	22%
GRE Tunnel MTU=9000	978 Mbits/sec	30%

4. 虛擬網路與主機整合情境案例探討

在本節我們將展示 Open vSwitch 結合 KVM 實作出各種虛擬主機與網路配置的案例。當使用者啟動虛擬機器時，透過動態配置 GRE Tunneling 的技

術讓虛擬機器間依客戶的需求以達到網路互連的目的。

4.1 情境一：兩部實體主機間新增多台虛擬主機

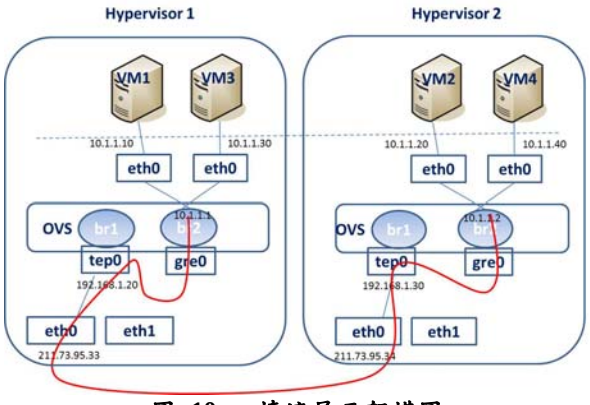

圖 10、 情境展示架構圖

在此情境中，我們延伸第 3 節的環境，原先是兩台虛擬主機分別位於兩部實體主機內部，透過底層虛擬交換器間的隧道互通，此時若配置多台新的虛擬主機，則僅需在實體主機內將這些虛擬主機與虛擬交換器互連即可，如圖 10 所示，此測試環境中，包含兩台實體主機分別為 Hypervisor1 和 Hypervisor2，每台主機下各執行兩個虛擬機器分別為 VM1、VM3 和 VM2、VM4，並屬於同一個網段，模擬經由相同的 bridge(br2)，透過建立專屬的 GRE Tunneling(gre0)讓四台虛擬機器彼此間網路互聯。實際上就像經由兩個 bridge 讓四台虛擬主機讓網路達到互連的目的，如圖 11。

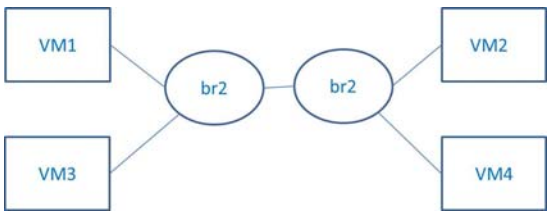

圖 11、 情境一虛擬連接拓樸圖

4.2 情境二：多台實體主機間之隧道互連

上述情境都是在兩台實體主機間建立點對點之隧道連線，若使用者的虛擬主機散佈於三台實體主機上，此時我們就需要建立多條隧道以串連這些實體主機，如圖 12，此測試環境中，包含三台實體主機分別為 Hypervisor1、Hypervisor2 和 Hypervisor3，每台主機下各執行一個虛擬機器分別為 VM1、VM2 和 VM3，並屬於同一個網段。其中 VM1 和 VM2 彼此模擬經由 OVS bridge(br2)，透過指向對端之 endpoint(tep0)所建立的 GRE Tunneling(gre0)讓兩台虛擬機器達到網路互聯。同樣的，VM2 和 VM3 彼此模擬經由 OVS bridge(br2)，透過指向對端之 endpoint(tep0)所建立的 GRE Tunneling(gre0)讓兩台虛擬機器達到網路互連目的。這裡，在 Hypervior2 上 GRE Tunneling 連接使用之 endpoint(tep0)，可同時提供給上述建立之兩條 GRE Tunneling 使用，毋須額外新增兩個 endpoint。就像 VM1 和 VM2 可經由各自的 bridge(br2)互連，VM2 和 VM3 經由各自的 bridge(br2)互連，而 VM1 和 VM3 會經由 VM2 之 bridge(br2)將彼此串起來，讓三台虛擬主機網路達到互連的目的，如圖 13。

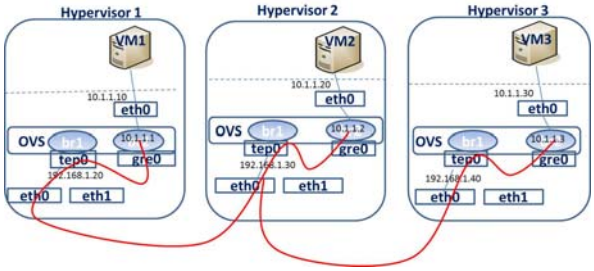

圖 12、 情境二展示架構圖

從圖 13 可看出本情境是將三個 br2 虛擬交換器透過兩條隧道互連，使用者亦可選擇建立三條隧道讓三個 br2 成為完全互連（fully-mesh），但如同第一節所述，第二層交換所使用的擴張樹路徑搜尋法無法有效運用所有的網路連結，在非擴張樹路徑上的連結均處於封閉狀態，待有斷線等狀況發生後才有可能被啟用。

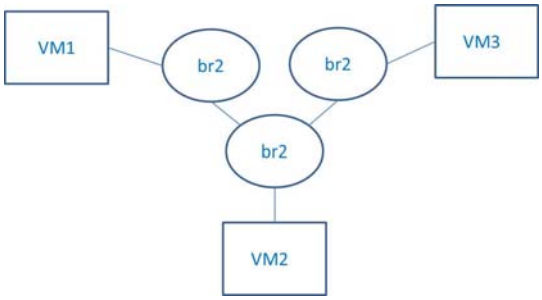

圖 13、 情境二虛擬連接拓樸圖

4.3 情境三：虛擬網路與實體網路互連

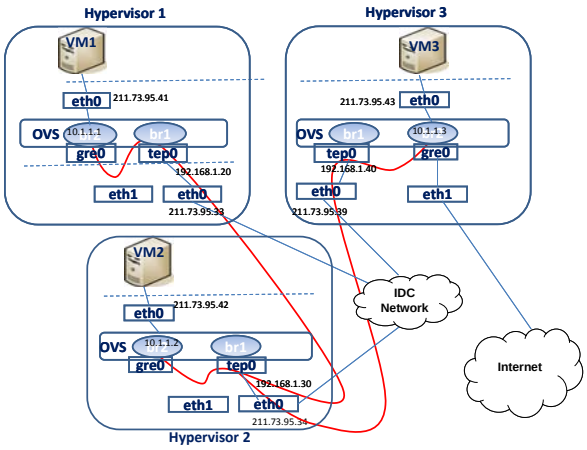

圖 14、 情境三展示架構圖

以上情境均為將虛擬機器置於私有網段內，如

使用者有與網際網路連接的需求，則我們需要在其中一部虛擬交換器與實體網路卡建立連結，以存取外部的網際網路，如圖 14，此測試環境中，包含三台實體主機分別為 Hypervisor1、Hypervisor2 和 Hypervisor3，每台主機下各執行一個虛擬機器分別為 VM1、VM2 和 VM3，假設均配置 Public 網段。我們模擬 Hypervisor3 上 OVS bridge(br2)將其設定使用之介面指定為 eth1(對外 public IP)，利用情境二之測試觀念，模擬讓 Hypervisor1 和 Hypervisor2 上之 VM1 和 VM2 可經由 Hypervisor3 上對外界接之介面(eth1)與外部 Internet 上之某一主機，達到對外網路互連目的，圖 15 為此情境之虛擬拓樸。

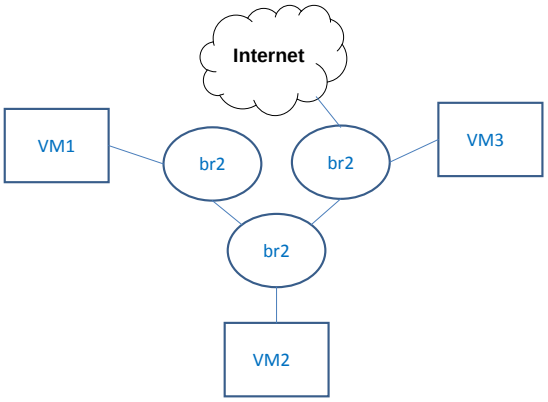

圖 15、 情境三虛擬連接拓樸圖

5. 結論

網路虛擬化技術最終需要與上層虛擬主機的整體配置整合，因此在發展的過程中除了既有網路設備商的規劃之外，尚要考量應用層的存取及操作的便利性，在網路與伺服器間尋求合作的過程中，雖然硬體網路設備嘗試著要主導網路交換的設定，但是在欠缺單一標準下讓使用者畏於投資，因此軟體交換器配合隧道技術隱然成為網路虛擬化的主流，然而隧道所需的表頭會造成額外的封包分割導致效能不彰，這部份也有賴將來新一代網路卡對於隧道封包的卸載功能提供解決方案，本論文整理了現有的隧道技術，並以實例方式對於各種可能的配置案例提供了佈建上的探討，可供整合虛擬主機與虛擬網路環境之參考；雖然網路設備的硬體式方案在此過程中未獲大量採用，但近年來興起的軟體定義網路 SDN 嘗試定義出網路控制的介面，統一的標準將可增加使用者採用之意願，屆時將可強化網路設備與伺服器間的合作關係，同時 SDN 與隧道技術的結合也將會是網路虛擬化下一個演進的方向。

參考文獻

[1] Ranjith Ramakrishnan, “What is cloud computing?,” CUMULUX, http://www.cumulux.com/Cloud Computing Primer.pdf

[2] J. Pettit, J. Gross, B. Pfaff, and M Casado, “Virtual Switching in an Era of Advanced Edges, “2nd Workshop on Data Center – Converged and Virtual Ethernet Switching (DC-CAVES), ITC 22, September 2010.

[3] J. Pettit, J. Gross, B. Pfaff, and M Casado, “Virtual Switching in an Era of Advanced Edges, “2nd Workshop on Data Center – Converged and Virtual Ethernet Switching (DC-CAVES), ITC 22, September 2010.

[4] Karsten Oberle, Marcus Kessler, Manuel Stein, Thomas Voith, Dominik Lamp, Sören Berger, “Network Virtualization: The missing piece,” ICIN conference, October 2009.

[5] IEEE 802.1Qbg - Edge Virtual Bridging, http://www.ieee802.org/1/pages/802.1bg.html

[6] IEEE 802.1: 802.1BR - Bridge Port Extension http://www.ieee802.org/1/pages/802.1br.html

[7] IEEE 802.1: 802.1Qbh - Bridge Port Extension http://www.ieee802.org/1/pages/802.1bh.html

[8] 曾惠敏、李慧蘭、胡仁維、劉德隆、張自恭、黃維誠，「使用雲端虛擬交換器進行網路虛擬化」，TANet2011 論文集，宜蘭，2011 年 10 月。

[9] VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks, Internet Draft, http://tools.ietf.org/html/draft-mahalingam-dutt-dcops-vxlan-04

[10] NVGRE: Network Virtualization using Generic Routing Encapsulation, Internet Draft, http://tools.ietf.org/html/draft-sridharan-virtualization-nvgre-02

[11] A Stateless Transport Tunneling Protocol for Network Virtualization (STT), Internet Draft, http://tools.ietf.org/html/draft-davie-stt-03

[12] The Overhead of Software Tunneling, Network Heresy http://networkheresy.com/2012/06/08/the-overhead-of-software-tunneling/

[13] Open vSwitch, http://www.openvswitch.org/

[14] Jumbo Frame, http://en.wikipedia.org/wiki/Jumbo_frame