國 立 清 華 大 學

碩 士 論 文

題目：政策性網路頻寬管理系統之研製

The Design and Implementation of Policy-based Network Bandwidth Management System

所別：資訊工程研究所

學號：874301　姓名：蔡汝霖

指導教授：黃能富 教授

中華民國八十八年六月

目錄

關鍵字對照表

BandKeeper	**頻寬管理器**
BandKeeper Agent	**頻寬管理器代理人**
File Transfer Protocol(FTP)	**檔案傳輸協定**
Hyper-text Transfer Protocol(HTTP)	**超文字傳輸協定**
Local Decision Point	**本地決策點**
MAC(Medium Access Control)	**媒體存取控制**
MIB(Management Information Base)	**管理資料庫**
PBMP(Policy-based Bandwidth Management Protocol)	**政策性頻寬管理協定**
Policy-based Network Management	**政策性網路管理系統**
Policy Consumer	**政策消費者**
Policy Decision Point(PDP)	**政策決策點**
Policy Enforcement Point(PEP)	**政策執行點**
Policy Maker	**政策制定介面**
Policy Management Tools	**政策管理工具**
Policy Rule	**政策規則**
Policy Rule Repository	**政策規則貯存點**
Policy Server	**政策伺服器**
Quality of Service	**服務品質**
Retransmission timeout	**重傳暫停時間**
Round trip time	**全程時間**
Sliding Window	**滑動視窗**
SNMP(Simple Network Management Protocol)	**簡易網路管理通訊協定**
TELNET	**遠端登錄協定**
World Wide Web(WWW)	**全球資訊網**

圖表目錄

第0章 導讀

近年來，隨著網際網路之迅速發展，各種網路相關的應用軟體也如雨後春筍般的出現在網際網路上。舉凡 World Wide Web(WWW)、Voice over IP(VoIP)、Netmeeting、Video conference、Video on Demand(VoD)、Distance Learning、MP3、FTP、Telnet、POP3、Games 等等。不同類型的應用軟體各有其特色與需求，有些屬於即時性的應用對於網路所造成的時間延遲相當敏感(例如 I-phone、Video conference 和 VoD 等等)，有些則側重資料傳輸的完整性(例如 FTP、Telnet 和 POP3 等等)。而當這些多元化的應用軟體在企業網路或是校園網路上同時使用時，往往會因為網路頻寬的不足而造成擁擠不堪的窘境，嚴重影響服務品質。例如當企業內部有人在網際網路上瀏覽網頁(WWW)或下載檔案(FTP)時，重要的 I-phone 可能因為無法使用足夠的頻寬而造成通訊品質不良。又例如校園中當網路遊戲(如 Mud)或網路音樂佔用太多頻寬時，正常的應用(Telnet、FTP 和 POP3 等等)也會受到嚴重影響。明顯地，因為網際網路用途的多元化，網路使用人口的增加，目前網路的頻寬實已達瓶頸。要疏解網路頻寬擁塞的壓力，擴大頻寬是最直接的方法，但這只是治標的短程解決辦法；最有效、最實際解決的治本方法就是將網路頻寬資源使用在真正需要使用的地方。

「政策性網路管理系統」提供高層次、集中式的控制管理理念及方法，我們希冀藉由這理念及方法，訂定「政策規則(Policy Rules)」，在當今複雜的網路環境當中，降低網路資源管理的負擔及成本。而現在很多的網路應用軟體，都是架構在 TCP/IP 上，著因於 TCP/IP 的一些特性，使得我們可以利用這些特性，配合「政策性網路管理系統」的機制及架構理念，來達到最有效的頻寬管制，以期在有限的頻寬下，能最有效益地使用頻寬資源。

本論文在第一章主要著重於「政策性網路管理系統」的架構及觀念介紹；第二章介紹構築在政策性網路管理系統上的頻寬管理系統之實作；最後於第三章則介紹測試環境及運作流程。

第1章 政策性網路管理系統

1-1. 簡介

「政策性網路管理」(Policy-based network management)已經成為現在最新的網路管理概念。很多廠商所研發的新一代網路相關產品，已經大量地利用這種「政策性網路管理」觀念來管理他們的產品，企圖在這種觀念的導向下，使網管人員可以很容易地管理現在越來越複雜的網路系統。政策性網路管理可應用的範圍包括服務品質(QoS)、網路安全(Network Security)、組態(Configuration)等等。

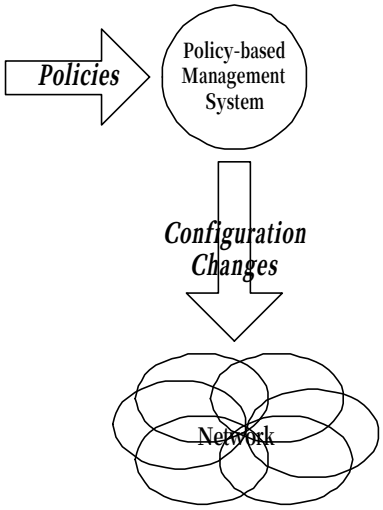


圖 1-1 政策性網路管理概念圖

如圖 1-1 所示：網管人員定義「政策規則(Policy Rule)」，經由「政策性網路管理系統」來限制、改變網路資源或服務，將在何時(When)、如何地(How)，被分配給誰(Who)使用[1]。政策性網路管理系統所扮演的角色就是將「政策規則」轉換(Translation)成設備特性之組態(Device-specific Configuration)，並且將該組態轉送至指定的設備，於是設備就得依據新給定之組態予以執行。

政策性網路管理系統呈現了一種新型態的管理：「將網路視為一實體，以進行管理，而不再類似傳統需對每台設備分別管理」。政策性網路管理和傳統的網路管理的關係就相當於高階語言和組合語言的關係。政策性網路管理和高階語言皆側重在「結果」，而不是在可以導致「結果」的「設備組態」之設定[1]。因此，

政策性網路管理系統提供一種自動化、人性化的機制來達到自動管理及自動組態的功能。本篇論文所要探討的是如何利用政策性網路管理的觀念，應用於服務品質，以達控管頻寬之目的。

1-2. 管理概念

政策性網路之管理概念，可抽象地分為三層，每一層中的每一元件並不一定要出現在政策性網路管理系統中，甚至某元件可能被封裝至另一層的另一元件中。政策性網路管理概念的三層約述如下：

1. 將一般設備或是資源之組態、操作特性、服務層次目標之管理具體化成網管人員可視的型態，這是系統的第一層所需具備的功能，而此功能由是由政策管理工具來提供。政策管理工具提供有意義且具親和力的使用者介面來呈現各種可管理之物件。而介於高階物件和低階設定之間，也需要有轉換或對應的能力。
2. 政策規則提供一詳盡的資源管理行為之集合。政策規則通常由政策管理工具所產生，貯存於政策規則貯存點，被政策決策點當作決策之依據，被執行於政策執行點。然而，在政策規則貯存至政策規則貯存點時，有可能會先直接至政策伺服器以作進一步的處理，例如，在政策規則由政策管理工具所產生之後，在貯存至政策規則貯存點之前，可能會先經由政策伺服器確認政策則規間沒有衝突。
3. 政策機制(Policy mechanism)是政策執行點的執行機制之陳述。政策機制是經由 APIs、方法(Methods)、協定(Protocols)或其它用來傳遞、估計政策及完成請求的函式來定義。最終地，政策機制導引政策執行點，執行網管人員經由政策制定工具所制定的政策規則，以完成請求。

1-3. 網路架構

如圖 1-2 所示，一套完整的政策性網路管理系統包含了四大部份[2][3]：

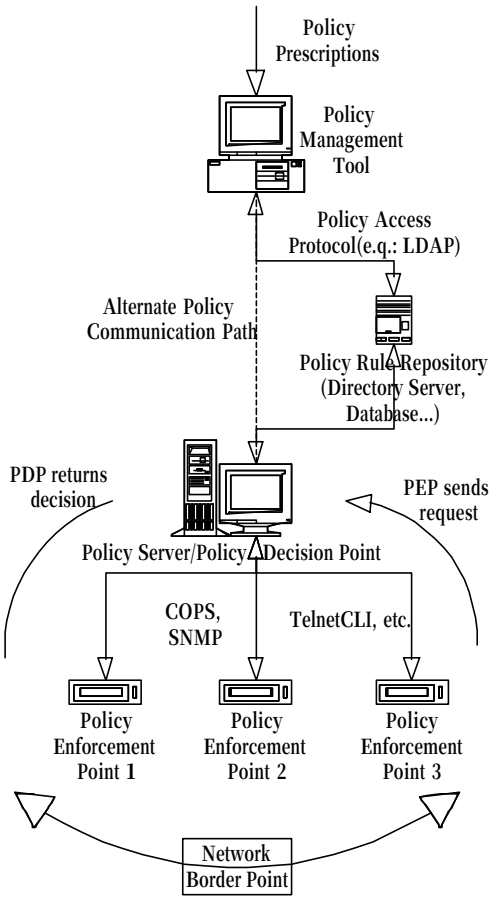


圖 1-2 政策性網路管理系統之架構圖

- **政策管理工具**(Policy Management Tools)
- **政策規則貯存點**(Policy Rule Repository)
- **政策伺服器/政策決策點**(Policy Server/Policy Decision Point/PDP)
- **政策執行點**(Policy Enforcement Point/PEP)

此四部份和於前一節所述的三層政策性網路之管理概念，是有一對應關係的：政策管理工具對應至第一層，政策規則貯存點和政策伺服器/政策決策點對應至第二層，而政策執行點則對應至第三層。以下就針對每一部份作一詳盡的功能介紹。

1-3.1. 政策管理工具

政策管理工具提供可對政策性網路管理系統進行管理的管理套件。以下是政策管理工具所需支援的功能：

- **政策編輯器/政策規則之呈現**(Policy Editing/Policy Presentation)**：政策編輯器提供一種可視、可輸入及可編輯貯存於政策貯存點之政策規則的機制和工具。這種機制和工具可讓網管人員使用來訂定政策規則，政策編輯器可能是以**Web**介面、圖形使用介面**(GUI)**或是命令列**(Command Line)**的模式來呈現。**
- **政策規則轉換**(Policy Translation)**：將服務層次目標**(Service Level Objectives)**轉換成政策服伺器可接受或解譯之政策規則型式，也就是將高階層次的政策之呈現，配合相關參數和屬性，以轉成政策規則給政策消費者使用。**
- **政策規則合法性**(Policy Validation)**：在編輯政策規則設定之時，兩種政策規則的合法性是需要檢查的：一是每個政策規則欄位的資料型態是否合於規定。如欲輸入 IP 的，就不能是負數型態；另一政策規則的語意**(semantics)**是否合於文法。如這樣子的一條政策規則，"從電腦** 10.23.24.56 **至電腦** 10.23.24.56 **須保證有** 50Mb/s **的頻寬"，這政策規則的每個欄位都無誤，但在語意上就出現矛盾。**
- **政策規則之衝突偵測**(Conflict Detection)**：每一條政策規則在政策執行點上都要能夠正確精準地被執行，所以絕對不能有政策規則之間的衝突或混淆發生。倘若有衝突或混淆發生，也須通知使用者以作進一步的處理。如：假設網路可利用之頻寬有** 10Mb/s**，定義如下的兩條規則：**
  政策規則 1: **從電腦** 10.2.24.56 **至電腦** 10.2.25.1 **須保證有** 7Mb/s **的頻寬。**
  政策規則 2: **從電腦** 10.2.24.58 **至電腦** 10.2.25.2 **須保證有** 5Mb/s **的頻寬。**
  這是可以很容易被察覺的。但並不是所有的衝突都可以在編修的時期察覺的，如果政策制定的欄位之中，含有動態的資訊(如時間)，就得等到執行時期才能偵測到，而在執行時期偵測這些動態的資訊，常常也是滿困難的課題。

1-3.2. 政策規則貯存點

政策規則貯存點是一個可以永久貯存政策規則及其相關資訊的存放點。在政策規則被確認其合法性並轉換成政策伺服器可接受格式之後，將會把政策規則貯存至政策規則貯存點，而這個動作可能發生在政策消費者(即政策伺服器)處理政策規則之前也可能之後。而當需要維護或更改系統裝置的狀態就需得從政策規則

貯存點將政策規則取出(Retrieval)。

1-3.3. 政策伺服器/政策決策點

政策伺服器[1]主要提供政策管理工具和政策執行點的溝通橋樑。以下是政策伺服器所需支援的功能：

- 政策規則更動通知：位於政策規則貯存點中的政策規則不時地被政策管理工具改變，政策伺服器需要有能力知道這改變，並且需通知政策執行點執行新的政策規則。
- 定位政策執行點：當接受來自政策管理工具的政策規則後，政策伺服器需能夠找出這些政策規則關聯的政策執行點，以提供足夠的資訊做規則衝突偵測並且也知道政策規則將應用至那些政策執行點。
- 決策能力：當政策執行點在執行政策時，在某些事件發生的情況下(如當 Host A 連往 WAN B 時，需得知道有那些資源使用限制)，此時，政策執行點會向政策決策點發出決策的請求，政策決策點需得依據政策規則做決策。

為達集中式管理，政策伺服器需具備有管理多個政策執行點的能力，如此可拓展政策性網路系統之延伸性，甚至在政策執行點之間也可發展出合作能力以執行管理工作。

1-3.4. 政策執行點

政策執行點是政策規則執行的地點。政策規則定義的種類可能有：服務品質保證，網路安全、頻寬整形…等等。政策執行點也就需要合適的政策規則，才能將政策規則正確的付諸於執行。但不論政策執行點需要什麼種類的政策規則，適時的記錄政策執行結果是相當重要的，這對於日後的問題追縱是很有幫助的。

1-4. 政策決策點和政策執行點之運作模式

政策伺服器和政策執行點之間的運作模式可分為兩種[4]：

- 委外處理模式(Outsourcing model)
- 直接運作模式(Provisioning model)

[1]政策伺服器又名政策消費者(Policy Consumer)。

委外處理模式是當政策執行點在某些情況(如連續好幾次的登錄失敗，這可能是有人嘗試破解或侵入)，需要進行「決策」，以讓政策執行點可以知道：它需要如何處理這種情況，於是向政策決策點送出「要求決策」的請求，政策決策點依據網管人員訂定的政策規則給予「決策」。

直接運作模式則是指政策伺服器接受來自政策管理工具的請求，將政策規則轉換成設備特性之組態規則，直接將組態規則傳送至政策規則執行點，政策執行點則依據政策伺服器所給定的設備特性之組態規則，付諸於執行，而不再向政策決策點請求「決策」。

在直接運作模式中，政策規則轉換成設備特性之組態規則並傳送至政策執行點。我們可將這模式做這樣的想像：在政策執行點的架構中，包含有「本地決策點」(Local Decision Point)，任何所需的決策直接由本地決策點完成決策，這類似將政策決策點中的決策功能移往政策執行點中。如圖 1-3 所示。

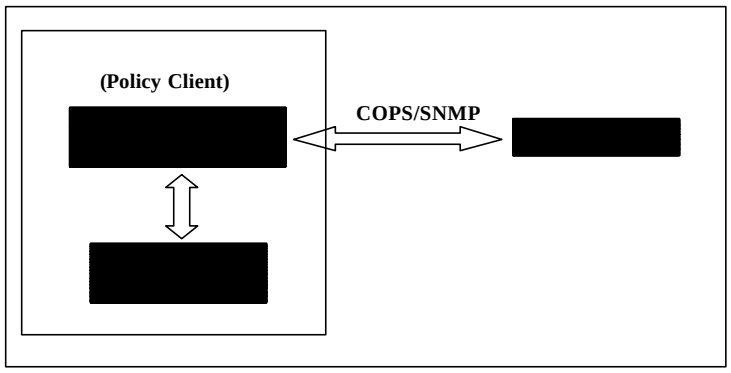


圖 1-3 政策執行點之本地決策點

在這裏，要強調一點，上面我們將政策性網路管理系統分成政策管理工具、政策規則貯存點、政策伺服器/政策決策點及政策執行點四大部份，這是從邏輯上來區分的；但或許在實際的實作上，可能會因某些考量因素而將某些部份做一些整合和變動。

1-5. 政策規則

網路管理員藉政策管理工具產生政策規則，政策決策點依據政策管理規則來適當地影響政策執行點的行為。政策規則定義服務和資源分配的邏輯。政策決策

點解譯政策規則，甚至更進一步確認政策規則的合法性(含規則間的衝突)，然後將其轉換成政策執行點可接受的組態。

政策規則的組成如圖 1-4 所示，如以語言的格式來表示[5]，即為：

if <*Condition*> then <*Action*>

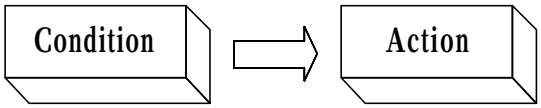


圖 1-4 政策規則組成型式

<*Condition*>可能是由下列這些屬性所組成的複合運算：

- Hosts：含來源端、目的端之位址範圍(如 Netmask)或是工作站之 IDs(如 DNS)。
- User：可識別之使用者或是接受端、傳送端可識別之 IDs。
- Applications：來源端、目的端之通訊埠範圍，傳輸協定(如 TCP、UDP、ICMP 等)。
- Layer2：來源端、目的端之 MAC 位址範圍、Ether type、802.1Q VLAN identifier、SNAP header 值、DSAP 或 SSAP 值等。
- Routing：依 IPv4/IPv6 位址介面、封包繞徑方向等。
- Schedule：周期性的時間(如每小時、每天，每星期、每個月等)等，允許周期時間屬性的指定，使得政策規則的訂定更具「因時制宜」的特性。

<*Action*>是指當<*Condition*>成立時，所需執行的動作程序。包含有服務的接受(Accept)、拒絕(Deny)、丟棄(Discard)，資源保留(Resource Reservation)及服務品質(QoS)的保證等等。

我們依照上面規策規則的定義方法，討論以下兩條規則可能發生的情形：

Rule 1:

```
if (SrcAddr == 10.114.1.2) then
   Drop packet.
```

Rule 2:

```
if (DstAddr == 10.113.2.2) then
   Forward packet.
```

Rule 1 定義的政策規則：「當封包來自於工作站 10.114.1.2 時，則丟棄它」；而 Rule 2 定義的政策規則：「當封包將往工作站 10.113.2.2 時，則轉送它」。在這兩條政策規的定義下，所有從工作站 10.114.1.2 往工作站 10.113.2.2 的封包就都可以吻合(match)這兩條規則，那到底要用那一條規則來處理這個封包呢?於是這就會產生「混淆」，這將可以怎麼解決呢？引進「優先權」的觀念就可避免掉這種不清楚的規則定義。假設 Rule 1 的優先次序大於 Rule 2，則當發生都可以吻合這兩條政策規則的情況時，則就以 Rule 1 為吻合的規則。

政策規則的定義是非常重要的關鍵，高度之可讀性、結構分明和層次分明，都可裨益網路管理員來訂定政策規則，以降低管理成本，減輕網管人員負擔。

1-6. 實作標準

目前，已經有很多的標準通訊協定已經被利用或整合至政策性網路管理系統：如 IPSec 用在確保政策決策點和政策執行點之間的安全通訊、利用 COPS (Common Open Policy Service)在政策決策點和政策執行點之間傳送請求和決策[6]、或是利用 SNMP 協定加上新增之 MIBs 來達到政策決策點和政策執行點之間的通訊[7]，和及正在發展的標準以 LDAP 來描述政策規則之方法[8]等等。

第2章 政策性網路頻寬管理系統

我們參照上述的政策性網路管理系統，以之理念來實作一套「政策性網路頻寬管理系統(Policy-based Network Bandwidth Management System)」。這套頻寬管理系統主要是針對 TCP/IP 之連接導向(Connection-Oriented)的服務品質來進行頻寬資源管制，即已經可辨識至 OSI 網路模式之第四層 –應用層的能力，除了達到頻寬管制的功能之外，尚有如下的功能及特色：

- 近端至遠端(Local to Remote)和遠端至近端(Remote to Local)之雙向(bi-direction)流量控管
- IP 和 MAC 配對，以防 IP 之盜用
- 可限制近端每台工作站，所能使用的最大流量之配額(Quota)
- 詳細記錄每條 TCP 連線的完整資訊，以利資料的再加值
- 跨平台管理工具的設計
- 可動態地更改政策規則，且不影響既有的連線

其中第二和第三點特色，特別能夠受到學校的青睞。就清華大學而言，宿舍 IP 盜用其實很泛濫的，若學校在學生申請使用宿舍網路時，就嚴格要求學生將 MAC 位址誠實登記，然後再透過頻寬管理器就可成功地制止那些愛盜用 IP 來隱藏自己來源的同學。

2-1. 設計的考量

在設計政策性網路頻寬管理系統時，我們做了如下的考量，為的就是讓網管人員更方便的管理。

2-1.1. 跨平台的設計

網路系統管理員可能在任何作業環境上操作，而我們不能限定唯一的操作環境，因此直接關係到網管人員的管理工具必須要能夠突破平台的限制。設計上，我們是以 Java Applet 作為實作工具，配合 Windows NT 平台的政策伺服器及 HTTP 伺服器來達成這遠端管理的目的。但重要的一點，由於 Java Applet 因為安全性的考量，所以它不允許直接連線到該 HTTP 伺服器之外的主機，因此政策伺服器和 HTTP 伺服器必須位在相同工作站上。

由於網路管理人員可以位在於我們將控管的任何地方(如 LAN 或 WAN)，它有可能因政策規則設定的方式而受影響通訊。最不被預期的情況是因為不當的訂定規則而被暫停(Block)住管理工具和政策伺服器的通訊。所以，Java Applet 的通訊方式，應該要有能力不受政策規則的影響。

2-1.2. 集中式控管

政策伺服器，應該提供可以對多台頻寬管理器控管的功能，藉由政策制定介面或是政策伺服器的幫助，甚至在所有的頻寬管理器之間，還得能夠分享、共用組態，設定、資源或政策規則，以增加其延展、延伸性，為達此目的，政策伺服器要和政策制定介面要有很多的配合。

2-1.3. 遠端管理

在頻寬管理器的網路設備中，想要達成遠端管理，就需要藉助於網路層的通訊協定來達成。我們採用自己私有的通訊方式 –PBMP(Policy Based Management Protocol)，此通訊方式是位於網際網路層(Internet Protocol, IP)之上的通訊協定。

網際網路是由數種透過路徑器及網路層協定的網路相連而成的，PBMP 能夠隱藏實體網路的連結，提供相同的介面服務。所以，每部頻寬管理器設備都要有 PBMP 代理人(Agent)來回應政策伺服器的要求，而 Java Applet 會透過管理伺服器來詢問或設定頻寬管理器設備。

2-1.4. 安全性

當有數個網路管理員直接使用政策制定介面同時管理時，可能會造成政策伺服器系統前後寫入不一致的問題。依我們設計的架構，政策伺服器能夠提供集中式的存取控制，對正在寫入的資源于以保護，並希望不久之後，能有能力拒絕服務某些非法的網路位址的連線而提供較佳的安全性。

政策性網路管理系統的安全至少需要用一個身份來鑑別(像政策管理工具需使用密碼登錄)。政策性網路管理系統包含很多需傳送之訊息和讀寫資料的部份。所有關於傳送訊息的互動，必需確定在一條安全的管道內進行傳送，包含政策管理工具和政策貯存點之間、政策貯存點和政策伺服器之間、政策伺服器和政策執行點之間的互動訊息，所以傳送端在傳送之前，得將請求或訊息加密；而接

受端在收到之後，先將請求或訊息解密並加以鑑認其合法性，若不合法(可能在傳送過程遭到篡改)，這需得提出警告並且予以記錄。

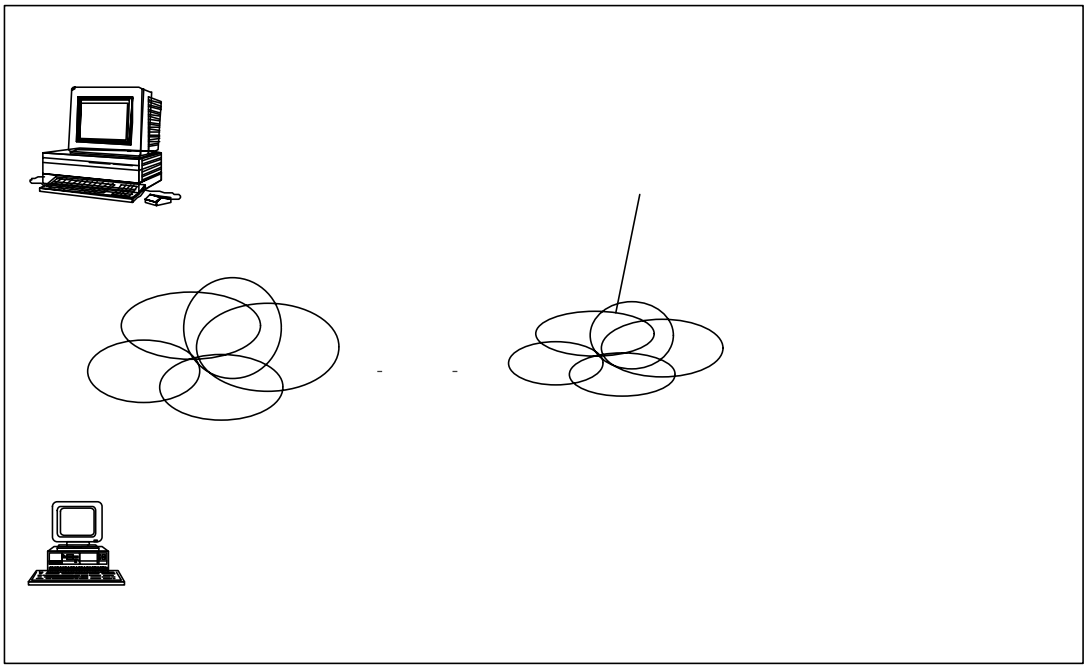
2-2. 網路系統架構

政策性網路頻寬管理系統主要包含了三個主要元件：

- 政策制定介面(Policy Maker)
- 政策伺服器(Policy Server)
- 頻寬管理器(BandKeeper)

我們已將政策貯存點實作至政策伺服器中。政策制定介面、政策伺服器及頻寬管理器三者之關係圖如圖 2-1 所呈現。政策制定介面是網際網路上的任一台工作站，透過瀏覽器(Browser)可以和政策伺服器連結，並取得管理的首頁，透過管理首頁，可啟動政策制定介面。政策制定介面是由 Java Applet 所寫而成，提供一非常具有親和力的圖形使用者介面(Graphic User Interface)。政策伺服器除了將使用者所要求的管理規則傳入政策制定介面所指定的頻寬管理器之外，並且也會接收來自頻寬管理器的各種請求，諸如網路事件，系統事件陷阱及連線記錄等等，並且會進行管理並分析連線記錄，以提供網管人員可以了解使用者在網路的使用行為分佈。政策伺服器同時可以控管多台位於不同網域的頻寬管理器，而不是單一的政策伺服器，只管一台頻寬管理器。

政策伺服器和頻寬管理器之間的運作模式，我們採用「直接運作模式」(Provisioning model)模式，政策伺服器直接轉送政策規則至頻寬管理器，讓頻寬管理器依據政策規則，自行決策並直接執行。



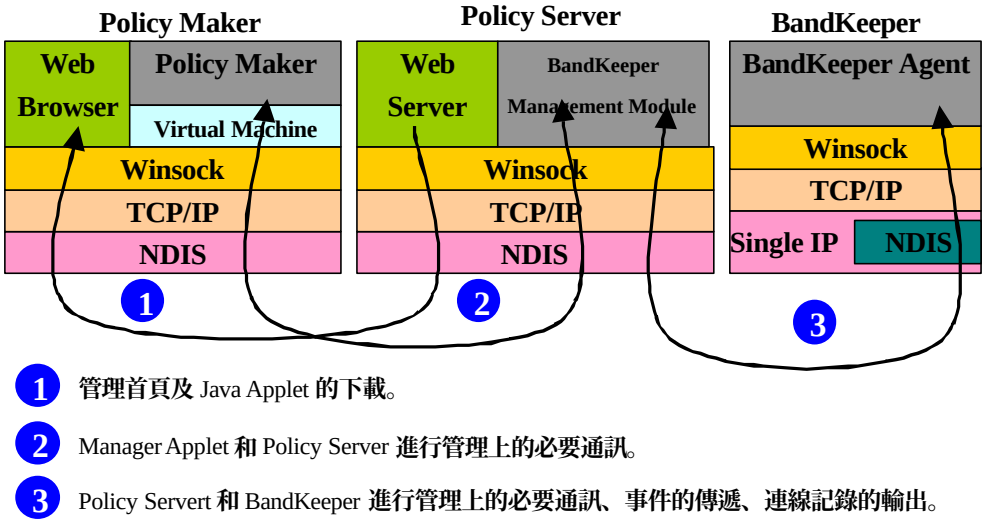


圖 2-2 Policy Maker、Policy Server 及 BandKeeper 的通訊協定堆疊

1. 政策制定介面和政策伺服器之間：管理者先用瀏覽器透過 HTTP(通訊埠 6592)將管理首頁從政策伺服器傳回，並從管理首頁啟動政策制定介面之 Java Applet。Java Applet 再利用 PBMP(通訊埠 6593)，進行設定管理規則所需的通訊(諸如 Download Books、Upload Rules、Discovery Device…等)。
2. 政策伺服器和頻寬管理器之間：在頻寬管理器上，有執行一 Task，其聆聽通訊埠 6594，接受來自政策伺服器的各種請求；政策伺服器也會從其通訊埠 6594 接受來自頻寬管理器的事件，連線記錄等。

PBMP 連線的請求方式，採取「一條連線，一個問題，一個結果」的「問-答」方式，如圖 2-3 所示。如此，很容易進行協定之設計，並且也容易進行除錯，但因 TCP 採用 3-way Handshake 的方式建連線，所以可能會效率不彰，不過對於發展的初期，這是比較好的選擇。因此每個問題就得有密碼的認証，而密碼通常是由可視的字元所組成，這就更顯出對請求和回應資料流(Data Stream)加密的重要性。

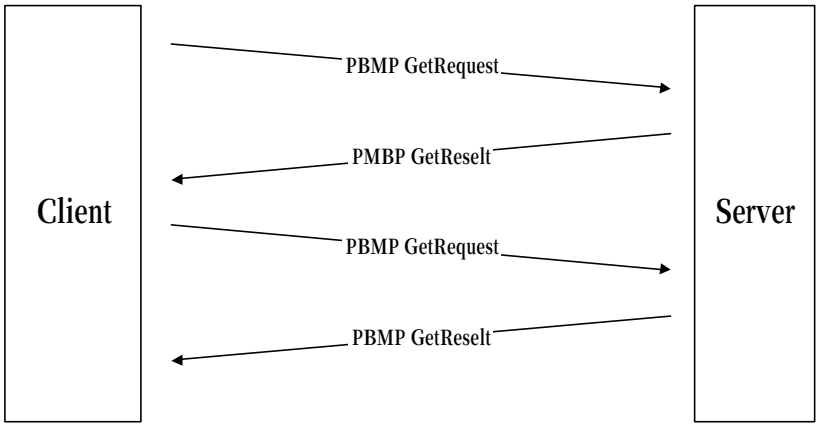


圖 2-3 PBMP 通訊示意圖

2-3.1. 加密與解密

加密，主要是增加資料的隱密性，為避免請求標頭(Request Header)和回應標頭(Result Header)在傳輸的過程，遭到不當的截取和修改。因此，在傳送端傳送請求或回應標頭（含政策伺服器和政策制定介面、政策伺服器和頻寬管理器之間）之前，都會進行加密；而當接受端收到請求或回應標頭時，就馬上進行解密並且稽核請求或回應的正確性及合法性，如果是稽核發生錯誤，就丟棄該次的請求或回應結果；若成功，才進行處理。有些請求或回應還會依需要而帶有酬載(Payload)，但我們對酬載就沒有進行加密。加密的原理採用簡單的邏輯運算，配合亂數產生種子，所以可以很快地完成加解密的動作。

2-3.2. 請求標頭和回應標頭之格式

在請求連線建立之後，請求端送出請求標頭，其標頭格式如下圖 2-4 所示：

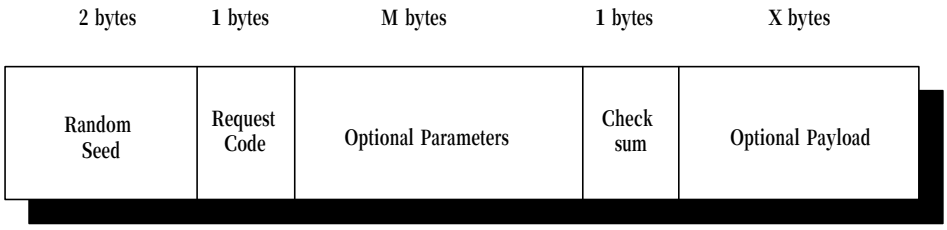


圖 2-4 Request Header 格式

Random Seed 主要是為加解密用，至於如何加解密，在此就不便多述。Request Code 指定請求代碼，每一種請求代碼會有不同的參數及酬載，Optional Parameters 和 Optional Payloads 即為指定這些額外參數及酬載。在政策制定介面和政策伺服器之間的請求標頭中的 Optional Parameters 之長度為 356 bytes；而在政策伺服器和頻寬管理器之間的請求標頭的 Optional Parameters 之長度則為 49 bytes，Optional Payloads 則會依請求代碼不同而有所不同的長度，而這個長度則由 Optional Parameters 指出。

而被請求端在處理完請求之後，便會送出回應。回應標頭之格式如圖 2-5 所示：

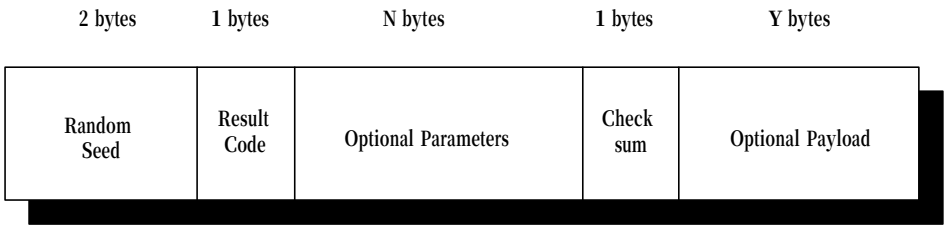


圖 2-5 Result Header 格式

Random Seed 也是為加解密用。Result Code 為回應代碼，除成功代碼外，對於每一種請求代碼，都有定義詳細之錯誤代碼。每一成功之請求，也會有不同的參數(Optional Parameters)及酬載(Optional Payload)，在政策制定介面和政策伺服器之間的回應標頭中的 Optional Parameters 之長度為 265 bytes；而於政策伺服器和頻寬管理器之間的回應標頭中的 Optional Parameters 之長度則為 32 bytes，Optional Payloads 則也會依請求代碼不同而有所不同的長度。

請求端送出請求標頭或被請求端送出回應標頭之前，必需將資料流以 Random Seed 編碼過，並填入正確的 Check Sum，才能經由 TCP/IP 送出。

2-3.3. 政策制定介面與政策伺服器通訊協定

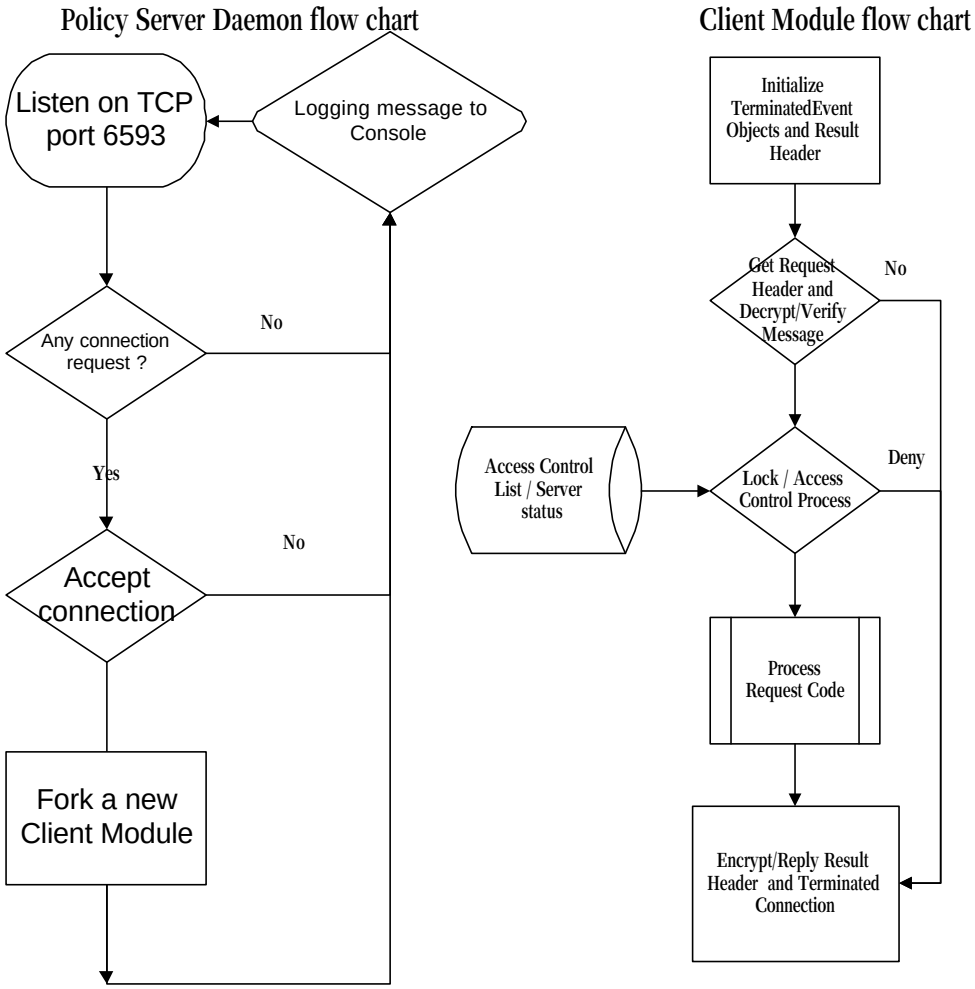


圖 2-6 政策制定介面與政策伺服器之通訊協定流程

政策伺服器 Listen 6593 **埠以等待連線，當政策制定介面向政策伺服器提出連線請求時，如果被政策伺服器接受，**Client Module **會被分支**(fork)**建立而出，以服務政策制定介面的請求。**

首先，Client Module **先建立** TerminatedEvent **物件，這個物件主要是用於在阻攔式**(blocking)**的** socket **中，能有接受外來中斷的事件。再來從** TCP/IP **連線資料流裏，將請求標頭讀入，然後進行解密、稽核並確認該請求為一已定義的請求代碼，再來檢查伺服器是否上鎖、驗証身份及密碼，若都無誤，再依請求代碼予以服務，將結果填入回應標頭，加密經由** TCP/IP **連線資料流，送回政策制定介**

面，並將連線中斷。

2-3.4. 政策伺服器與頻寬管理器通訊協定

基本上，政策伺服器與頻寬管理器之通訊協定流程和政策制定介面與政策伺服器之通訊協定流程很像，只不過頻寬管理器代理人聆聽 6594 埠。還有一點，頻寬管理器能指定唯一許可存取的政策伺服器，所以並非任一台政策伺服器可以和任一台頻寬管理器溝通。

2-4. 政策制定介面

政策制定介面為達跨平台，並提供管理遠端管理之目的，於是利用 Java Applet 來發展並提供圖形使用者界面(GUI)的設定介面、如圖 2-7 之呈現。政策制定介面提供以下的功能：

- 制訂頻寬管理的政策規則
- 確認政策規則的合法性
- 自動找尋頻寬管理器
- 可同時控管多台的頻寬管理器

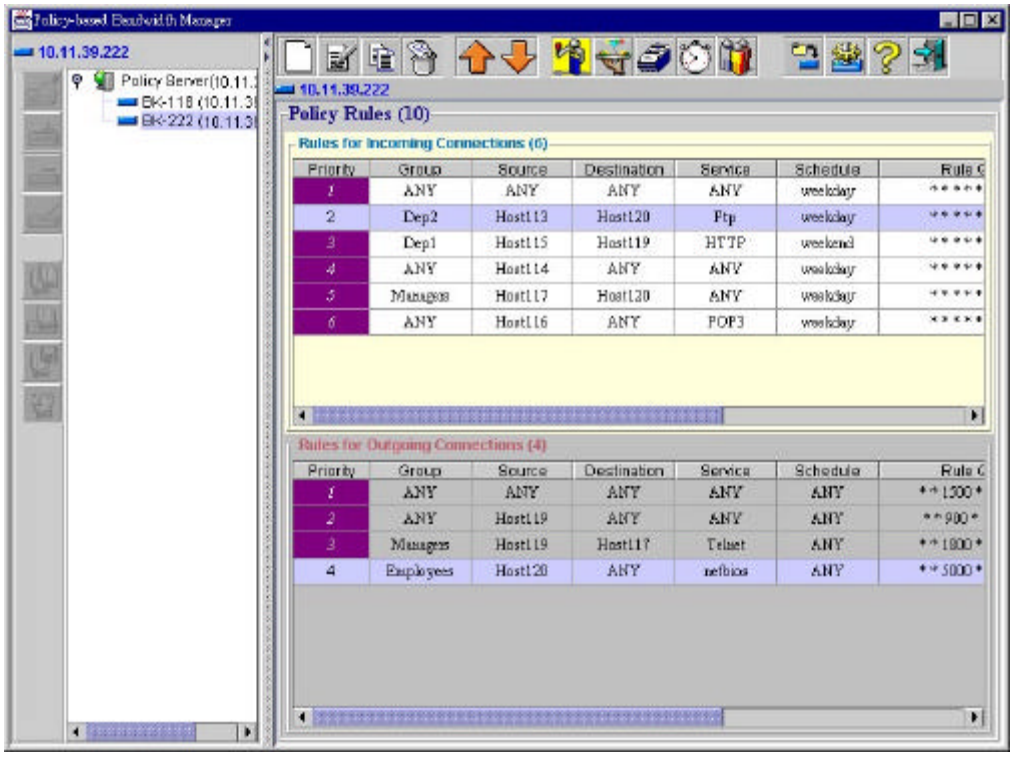


圖 2-7 政策制定介面之外觀

外觀上，這是很具親和力的管理介面。左半邊呈現出可被管理的頻寬管理器。而當要對一台全新的頻寬管理器進行管理前，需得進行以下的初始步驟：

1. 得知頻寬管理器的通訊字串(即 SNMP 之群組身份 Community)。
2. 初始頻寬管理器之組態及指定頻寬管理器將允許管理的政策伺服器。
3. 從政策制定介面將新的頻寬管理器手動加入或經由自動找尋頻寬管理器的功能加入。
4. 雙擊欲管理的頻寬管理器以初始及載入已存在的政策規則。
5. 進行制定政策規則。

右邊則呈現政策規則的組態，可提供網管人員進行編輯、修改等動作，完成編修後，按下「執行鈕」，政策制定介面先進行政策規則的合法性及衝突檢查，若無誤，則把所有的政策規則及相關資訊組態傳回政策貯存點(已併入至政策伺服器)貯存，並且將新的政策規則經由政策伺服器送往頻寬管理器執行。

2-4.1. 政策規則

我們將政策性網路頻寬管理系統的政策規則分為 Incoming(遠端至近端)和 Outgoing(近端至遠端)兩大部份。每一條規則由 Source IP/Netmask、Destination IP/Netmask、Protocol、Service、Schedule、Group 組成*<Condition>*的部份。*<Action>*的部份則由 Rule QoS 和 Connection QoS 來描述。Rule QoS 是控制所有符合政策規則的連線之服務品質；Connection QoS 則是用來控制個別連線之服務品質。QoS 包含有 Maximum/Committed Rate、Quota limitation、Connection Duration。

若政策規則有「優先權」的觀念，再配合上 Schedule 的性質，就可以形成「QoS 隨時間變化」的特性。如以下兩條政策規則：

Rule 1:

Condition:

```
Source IP = 192.168.168.1
Source Netmask = 255.255.255.255
Destination IP = 192.169.169.1
Destination Netmask = 255.255.255.255
Protocol = TCP
Service = FTP
Schedule = Morning Hour (6:00am~10:00am)
```

Action:

```
Committed = 1500 Kb/s
```

Rule 2:

Condition:

```
Source IP = 192.168.168.1
Source Netmask = 255.255.255.0
Destination IP = 192.169.169.1
Destination Netmask = 255.255.255.0
Protocol = TCP
Service = ANY
Schedule = Office Hour (8:00am~5:00pm)
```

Action:

```
Maximum = 3000 Kb/s
```

當從工作站 192.168.168.1 **連往工作站** 192.169.169.1**，做檔案傳送服務時：**

- **若連線時間在** Morning Hour(6:00am~10:00am)**，則依** Rule 1 **會有** 1500 Kb/s **的保證頻寬**。
- **若連線時間在** Office Hour(10:00am~5:00pm)**，則依** Rule 2 **會有最大** 3000 Kb/s **的頻寬**。

「QoS **隨時間變化」的特性，對於政策規則的訂定有很大的彈性，比如某些** ISP **的用戶向** ISP **提出這樣子的請求：「我只付上班時間**(8:00~17:00)**的網路頻寬使用費」，為要能滿足客戶的需求，只要設定好** Schedule**，就能使政策規則在適當的時候生效**。

在我們的設計中，使用者在定義政策規則之前，必須事先定義好 Address Book**、**Service Book**、**Schedule Book **及** Group Book**。在各種** Book **中，每一項目**(entry)**可以指定別名**(alias name)**，項目包含若干欄位，在輸入時，就會進行數值合法性的檢查；而真正在定義政策規則時，則從各種** Book **中，以下拉式選單「選擇」的方式來制定政策規則，如此一來因為網路管理員可以依照自己習慣，訂定易記名稱，這可以增加可讀性，二來可以達到「防呆」效果，這可以避免使用者定義出** Source **和** Destination **居然是同位在近端或是遠端的錯誤**。

2-4.2. 實作平台及發展環境

政策制定介面在 Windows **作業環境下，利用** JBuilder **作為發展工具，再配合新版的** JDK **及使用非常具親和力的** Swing **視覺化套件，發展而成的** Java

Applet，如此可跨越平台藩籬的限制。

2-5. 政策伺服器

政策伺服器介於政策制定介面與頻寬管理器之間，做為兩者的溝通橋樑。政策伺服器包含有如下的功能：

- HTTP 伺服器和政策規則貯存點
- 頻寬管理系統之管理模組
- 連線資料的再加值

HTTP 伺服器主要用來服務管理首頁及統計系統報表首頁，以讓網管人員可以從這些首頁進行管理和觀看統計資料報表。

頻寬管理系統之管理模組，又包含以下六個模組：

1. 頻寬管理器管理模組：負責頻寬管理器的組態資訊之設定、取得及政策規則之下載。
2. 政策伺服器管理模組：負責政策伺服器上鎖、解鎖及重置，避免因多重進入管理造成資料不一致和錯誤。
3. 登錄管理模組：主要提供政策制定介面，可以貯存政策規則及其它相關的登錄資料，我們將政策貯存點合併至此模組中。
4. 事件通知管理模組：集合政策伺服器及頻寬管理器產生之事件(如連線記錄無法送達政策伺服器、磁碟已滿無法記錄連線資料…)，依照事件等級，透過電子郵件通知管理者。
5. 連線記錄模組：專門接受來自頻寬管理器的連線記錄資料(包含有 Source IP/Port、Destination IP/Port、In/Out Octets、Start/End Time 等)。
6. 連線資料分析模組。

利用這些模組使得政策制定介面能夠完成政策規則的制定和執行政策規則。頻寬管理器在執行政策規則時，會把每一筆的連線記錄確實送回政策伺服器記錄存檔，政策伺服器會將這些連線資訊定時做統計及匯整。

2-5.1. 實作平台及發展環境

目前的政策伺服器的實作平台是在 Windows NT 4.0 而且以 Service Packet 6

更新(patch)過系統，不過，因為政策伺服器是架構在 Winsock 之上，所以在 Windows 2000 也可以正常的運作。既然是利用 Winsock 寫成，所以其實在 Win 32 的平台上，都可以正常的運作，但是我們選用具有 NT 架構(含 Windows 2000) 作為實作的平台有幾點的原因：

1. NT 有伺服器所必需有的特性 –支援 Service(類似 UNIX 的 Daemon)，我們總不會想看到我們的程式還得使用者以超級管理者的身份登錄 Windows 才能使用吧!
2. NT 有類似 UNIX 的 crontab 的指令 –at。可以指定在某個特定的時間時，做特定的事，這對於我們需定時進行資料分析非常有用。

2-5.2. 頻寬管理器管理模組

政策制定介面在對頻寬管理器的進行管理時，所需要用到的資訊，都得透過政策伺服器向頻寬管器取得，這些資訊包含有網路介面之速度、MAC 位址、政策規則之下傳等等，如圖 2-8。

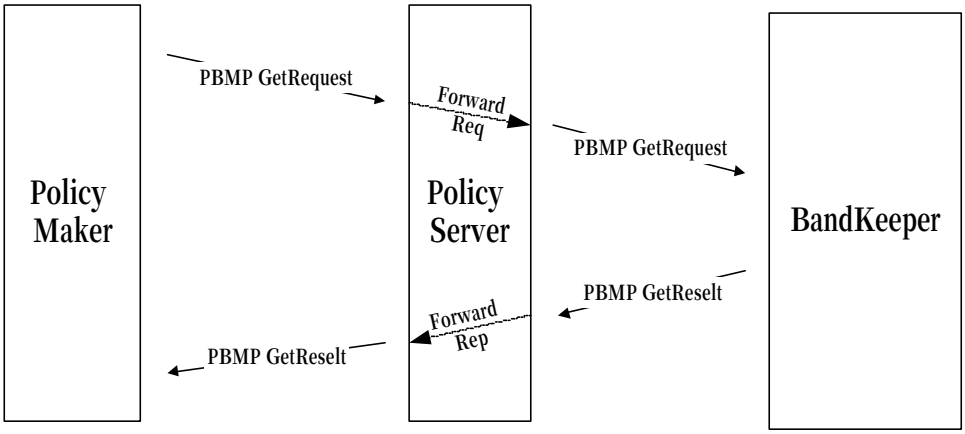


圖 2-8 頻寬管理模組轉送請求/回應示意圖

2-5.3. 政策伺服器管理模組

我們採用「一條連線，一個問題，一個結果」的方式進行政策制定介面和政策伺服器間的溝通，但為避免多個政策制定介面同時存取政策伺服器，造成資料之不一致性，於是我們就得避免這個情況的發生。

如圖 2-9 所示：

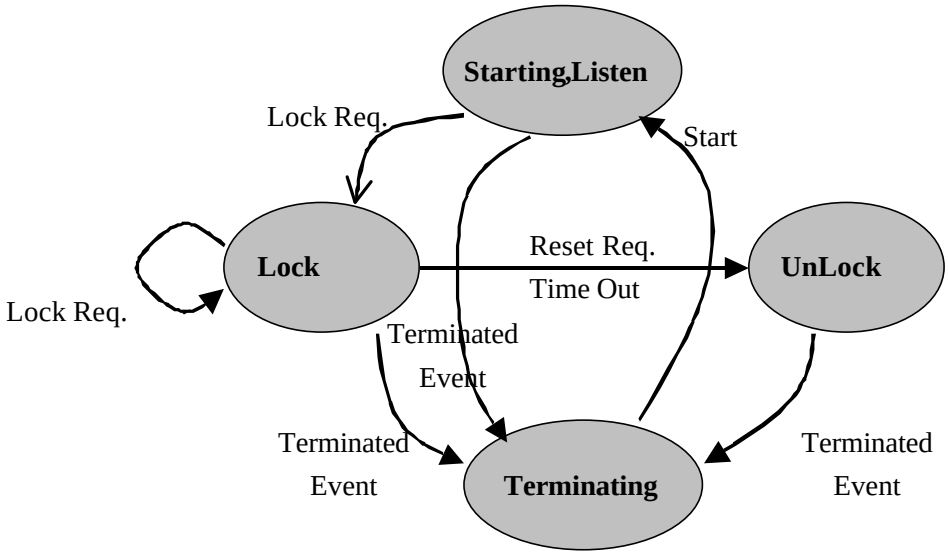


圖 2-9 政策伺服器的狀態機

當政策伺服器一旦被啟動，就處於 Listen **的狀態，等待政策制定介面連線的請求。在政策制定介面連上之後，就送出** Helo **之請求，在政策伺服器確認其身份後，再馬上送出** Lock **請求(此請求不會被斷線)，以開啟一個服務的** session**，並將政策伺服器** Lock **住，以避免第二台政策制定介面試圖連線，在往後的每** 120 **秒之內，政策制定介面必須傳送一可識別之** Signature **至政策伺服器，否則政策伺服器會因等待接收** Signature **的逾時而自我** Unlock**。不管政策伺服器在那一種狀態下，都得允許被中斷** session **的執行，因此，政策伺服器還得能夠接受** TerminatedEvent **事件的觸發而終止** session**。**

在政策伺服器處於 Lock **狀態的時候，只有下達** Lock Request **的政策制定介面才有能力進行和政策伺服器溝通的能力，除了** GetActiveConnection **和** Helo **這兩個請求之外，其他的請求皆受到來源的限制。**

2-5.4. 事件通知管理模組

事件的來源有二種，一是政策伺服器自發性的事件，這可能會由各模組運作時觸發，二是來自頻寬管理器之事件。我們將事件分成三種等級：錯誤(Error)、警告(Warn)、通知(Inform)。網路管理員可以定義欲接收的事件等級，在當事件發生時，就會經由電子郵件(E-Mail)收到通知。除了電子郵件通知之外，所有的事件也會在政策伺服器留下記錄，我們也實作另一檢視器可觀看歷史事件，如圖

2-10 所示。

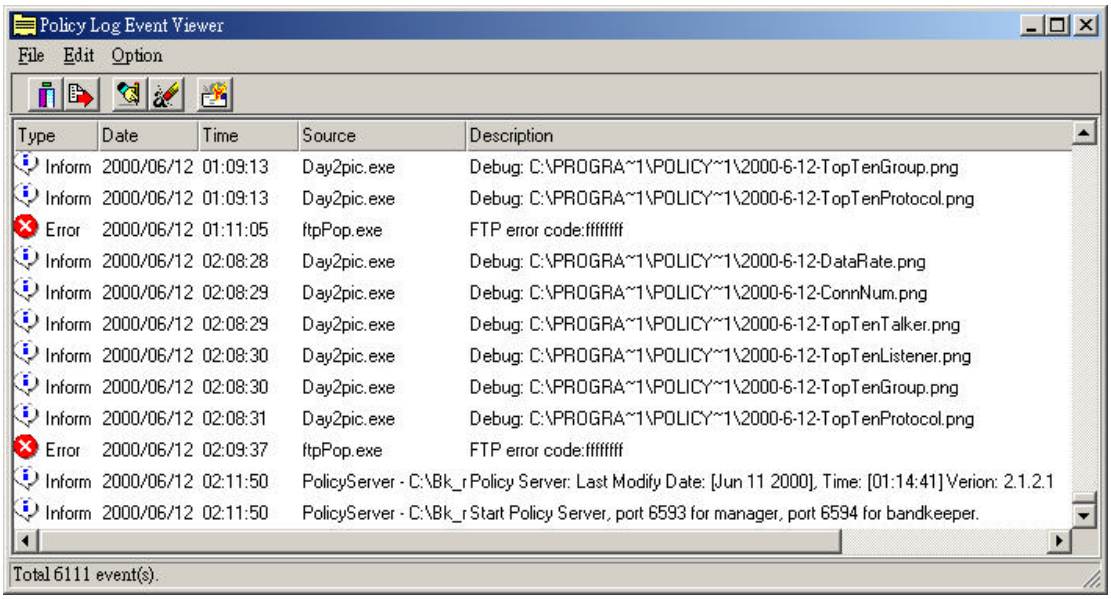


圖 2-10 政策性網路頻寬管理系統之事件檢視器

2-5.5. 連線資料分析模組

連線資料分析模組負責在定期的時間內，將連線記錄模組收集到之連線資料予以分析，以圖形化的介面呈現，可供網管人員觀察網路使用者之使用行為。分析的項目包含有每個小時之連線數目、傳送的總量，前十大 Group、Talker、Listened、Protocol **等，每一種項目，又有以小時計、天計、週計、月計和年計等統計資料，如圖** 2-11 **所示。由這些統計資料，可以很清楚地看出那些使用者流量最突出、那台伺服器最常被存取、網路最常使用的應用、…等。甚至進一步，可以經由尋找(search)的功能，可追縱出使用者歷史連線記錄。**

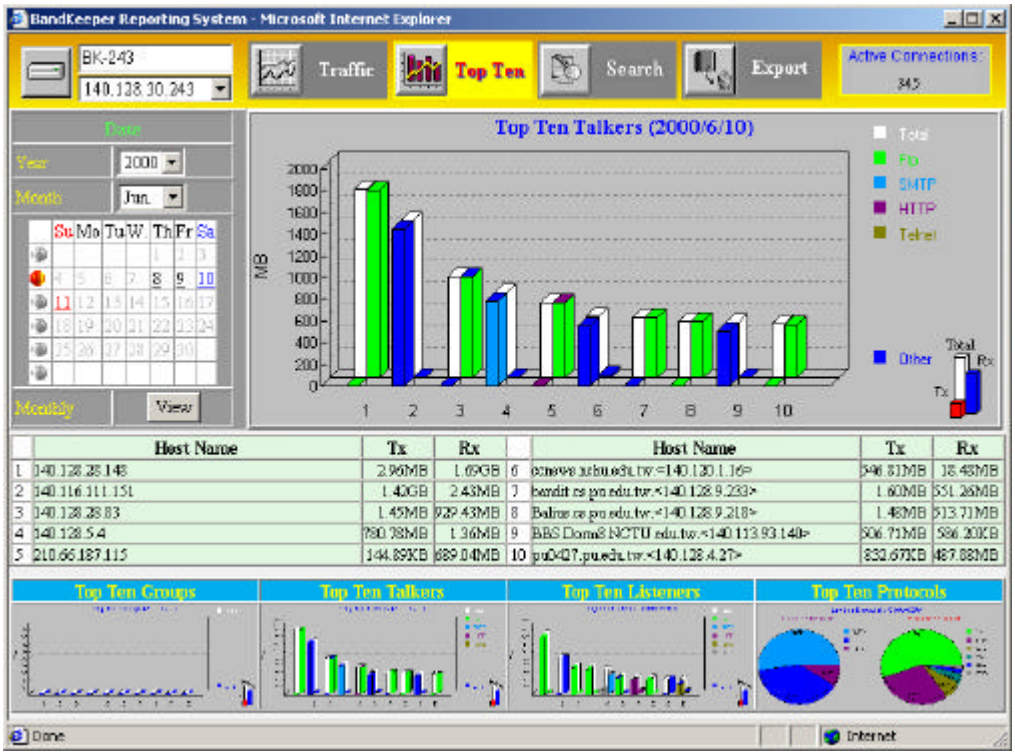


圖 2-11 連線資料統計圖

2-6. 頻寬管理器

頻寬管理器是政策性網路頻寬管理系統之政策執行點，主要工作是依據來自政策伺服器給定的政策規則以做流量的控管。頻寬管理器包含以下三大部份：

- 頻寬管理器代理人(BandKeeper Agent)：利用 PBMP 協定來負責和政策伺服器的所有通訊，包含從政策伺服器下傳政策規則及從頻寬管理模組取得連線記錄上傳至政策伺服器。
- 頻寬管理模組(BandKeeper Module)：依照網管人員制定的政策規則來對於每條穿越過頻寬管理器的資料流(flow)，給予適當之服務品質，並負責監控、記錄所有連線之資訊，包含有：來源端之位址及通訊埠、目的端之位址及通訊埠、通訊協定、政策規則的代號、流入及流出總量、丟棄流入及流出總量、起始及終止時間、終止原因等。
- 事件傳送模組(EventLog Module)：負責補捉發生的各種事件。

頻寬管理器置於既有的網路架構上，具有「透明」(transparency)的特性，也就是不會影響舊有的網路拓樸架構(topology)。但頻寬管理器會依照政策規則來

限制或保證頻寬並隨時監控所有穿越它的連線，而且具有辨識 OSI 網路模式之第四層 –應用層的能力，所以在政策規則的定義中，可指定諸如遠端登錄協定(Telnet)、檔案傳輸協定(FTP)、超文字傳輸協定(HTTP)等高層次的傳輸協定。

2-6.1. 頻寬管理模組

當政策規則經由政策伺服器下載至頻寬管理器後，如圖 2-12 所示，頻寬管理器模組將對於網路每一條 TCP/IP 連線的封包，從政策規則表(Rule Table)找出吻合之規則，以得知其應有的服務品質，並累積該連線之進出流量，計數連線時間等資訊。

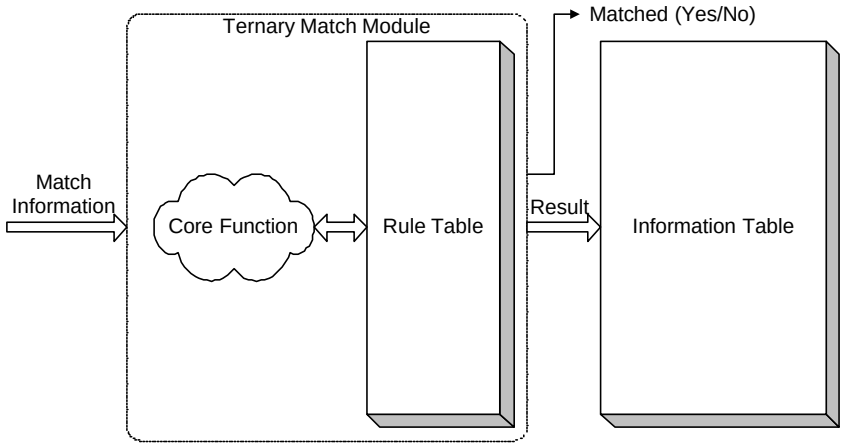


圖 2-12 頻寬管理模組之封包分類模組

頻寬管理模組為精確管制頻寬資源及達到最大使用效率，於是採用「Delay-Ack」配合改變「Sliding Window」大小的技術來控管頻寬。計算出來回傳送端和接受端的全程時間(round trip time)，攔截和延遲 TCP 標頭中的 Ack 旗標，在重傳暫停時間(retransmission timeout)之內，平順地控制頻寬。

當某條連線速度過快時，如圖 2-13 所示，因為 Ack 旗標被頻寬管理模組所延遲的時間，需小於傳送端和接受端之 TCP 重傳的時間，而當 Ack 旗標被頻寬管理模組傳送出去時，也送出更改 Sliding Window 大小[2]的要求，以降低連線的傳送速度。

[2] Sliding Window 的大小和速度的快慢有絕對的關係，藉由事先的量測可得知它們的對應關係。

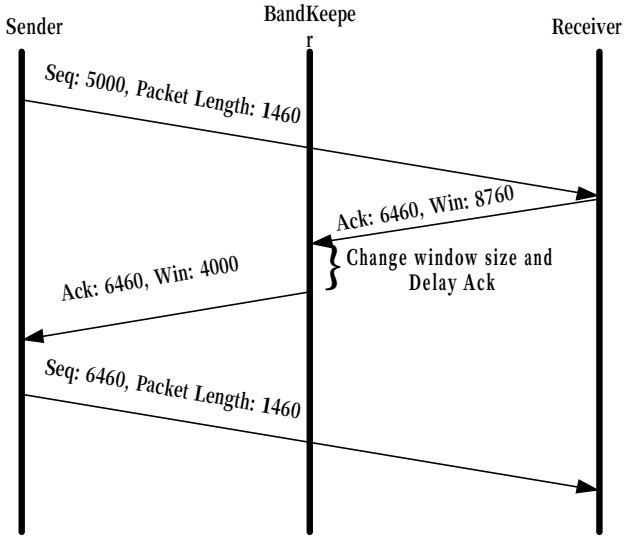


圖 2-13 頻寬管理模組控制過快的連線

相對的，當某條連線速度過慢時，如圖 2-14 **所示，因為** Ack **旗標被頻寬管理模組所延遲的時間有可能大於傳送端和接受端之** TCP **重傳的時間，而導致重新傳送，這將浪費掉網路頻寬，所以來自接受端的** Ack **旗標，應該馬上被傳送出去時，且此時** Sliding Window **設為零以暫停**(blocking)**傳送，等到延遲時間到，再重設** Sliding Window **大小以繼續傳送資料，這將可避免因為傳送速度過慢，造成的重傳浪費**。

明顯地，我們可以歸納出採用「Delay-Ack」**配合改變**「Sliding Window」**的技術有以下幾項的優點：**

- **更 有 效 益 的**
- **沒有封包在緩衝區等候**(Queue)
- **無須大量繁而雜的運算或演算法**
- **更流暢地控制頻寬**

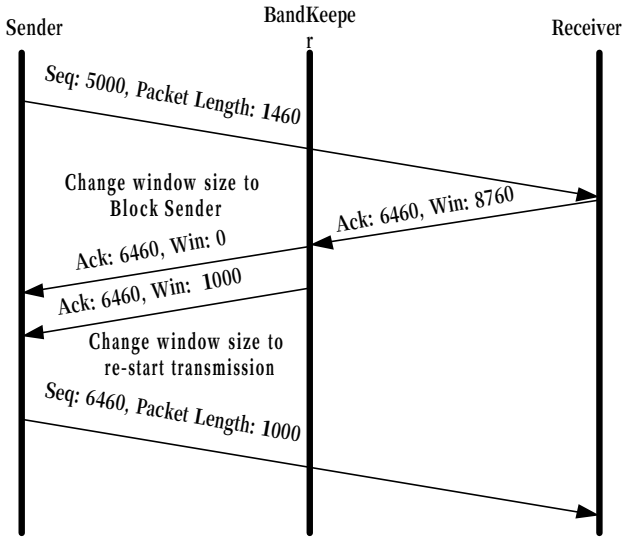


圖 2-14 頻寬管理模組控制過慢的連線

2-6.2. 頻寬管理器代理人

頻寬管理代理人主要用來回應政策伺服器的請求，可參照圖 2-8。**此外，它有個計時器**(Timer)，**會定時向頻寬管理模組取得連線記錄資訊，送回政策伺服器。計時器的觸發時間間隔是可設定的，過長將因為記憶體不足以儲存連線記錄資訊而導致連線記錄流失；過短將因常常地建立連線而降低效率，造成不必要的浪費。**

2-6.3. 事件傳送模組

此模組負責將頻寬管理器發生之事件，送至政策伺服器，再由政策伺服器依網管人員的事件通知設定及事件等級，經電子郵件通知網管人員。

邏輯上，此模組維護一個環狀的佇列(queue)，**當有事件發生時，就會先被送往此佇列，事件傳送模組會嘗試著在網路暢通時，再事件送至政策伺服器；若恰巧網路出問題而無法連至政策伺服器時，就會先留在此環狀的佇列，以等待網路恢復。而此環狀的佇列大小固定為** 512Kbytes，**佇列大小並不會隨著事件變多而變大，因此當佇列達到滿的狀態時，就會從最舊的事件開始覆蓋。這對於在嵌入式系統**(embedded system)**中，有限的貯存空間下，是很值得實作的一項機制。**

2-6.4. 實作平台及發展環境

頻寬管理器的發展環境是使用一部個人電腦，其中作業系統同樣也是 Winows NT 4.0 並且也以 Service Packet 6 更新過。個人電腦上插有兩片網路卡，各自負責對 WAN 和 LAN 之網路封包的收送，而受管制的就是 WAN 至 LAN 或是 LAN 至 WAN 的網路流量。如圖 2-15 所示：

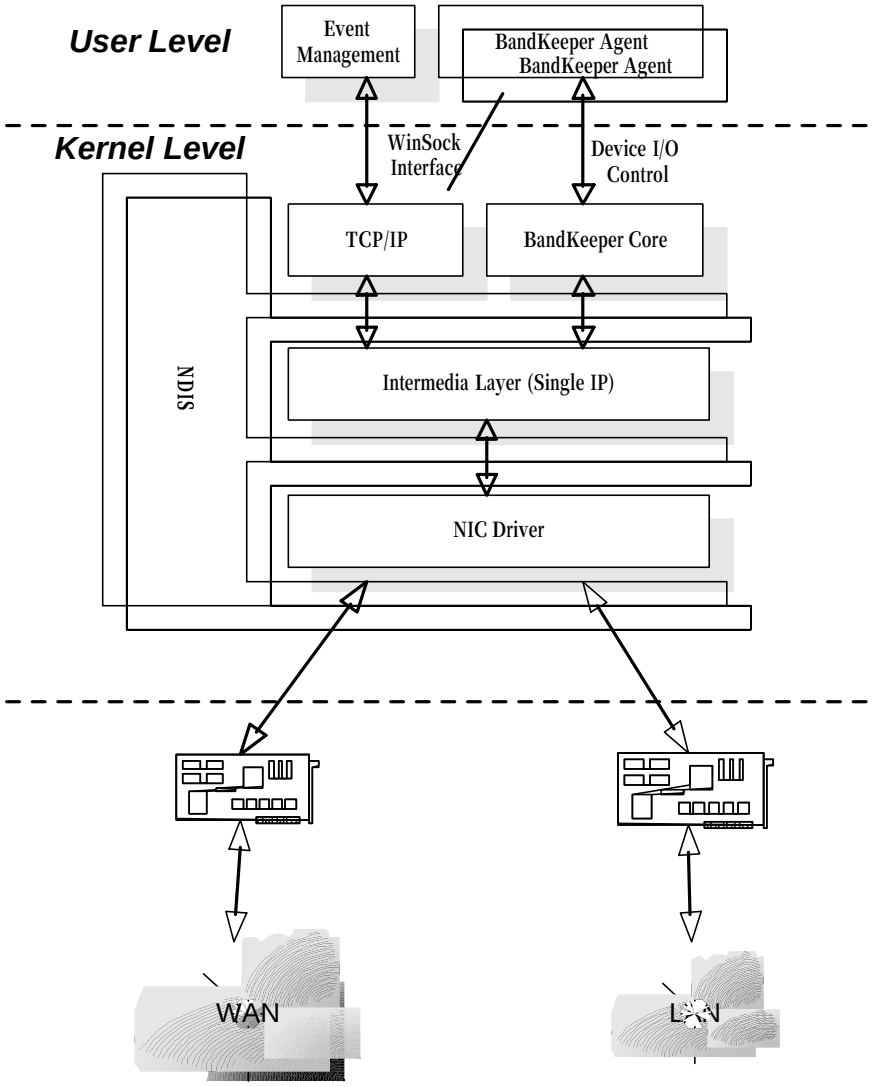


圖 2-15 頻寬管理器內部運作流程

因 NT 不允許一個 IP 被指定至不同的網路卡上，這將造成政策伺服器只能擺在 LAN 或 WAN 的其中一邊，但政策伺服器是可以同時控管多部的頻寬管理

器，如此很容易造成管理上的問題。假設政策伺服器只能位於頻寬管理器的 LAN 埠，如圖 2-16 所示，政策伺服器想要管理此二部頻寬管理器；從圖中可以看出政策伺服器位於頻寬管理器 1 的 LAN 埠，可以成功地和頻寬管理器 1 進行管理通訊；但因位於頻寬管理器 2 的 WAN 埠就會遇上無法成功的和頻寬管理器 2 進行管理通訊的窘境。

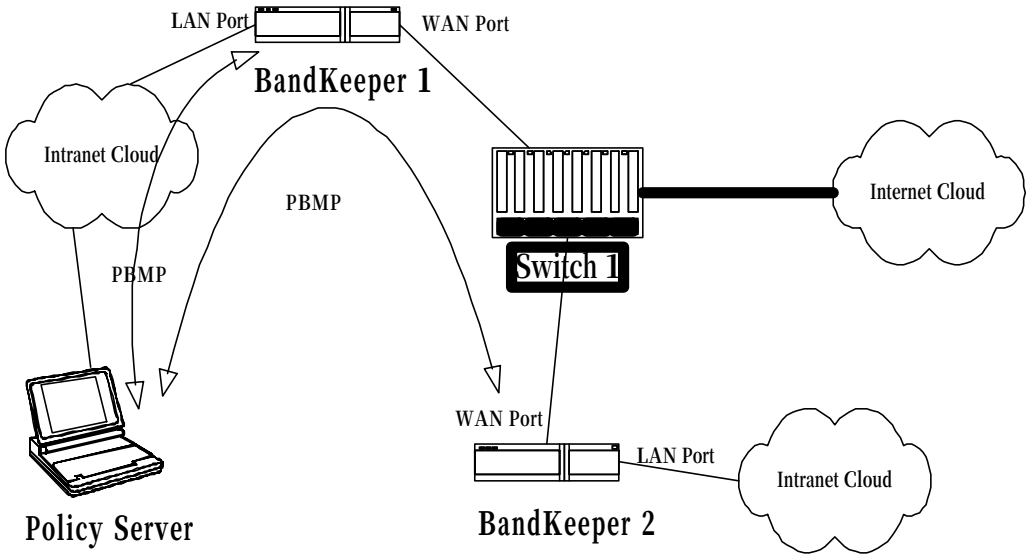


圖 2-16 政策伺服器管理多部頻寬管理器示意圖

為讓我們的政策伺服器不論位於 WAN 或 LAN，皆可和頻寬管理器代理人通訊，我們攔截 Intermedia Layer 以解決頻寬管理器代理人和政策伺服器之間的 TCP/IP 通訊，使得政策伺服器不論位於 WAN 或 LAN，皆可和頻寬管理器代理人通訊。此舉就好像造成在 NT 下，把同一個 IP 指定至兩張網路卡一般。

當有來自 WAN 或 LAN 的網路封包，經網路卡驅動程式接收，經 Intermedia Layer 判定，若是給頻寬管理器代理人，則經由 TCP/IP、Winsock 介面傳至頻寬管理器代理人；若非，則經由 NDIS 的通知，BandKeeper Core 就會依據政策規則進行分類，丟棄，轉送來控制頻寬資源。

2-7. 資料報表及加值

政策伺服器內建有小型之資料庫(與其說是資料庫，倒不如以固定區塊的檔案形容會更貼切)，可暫時儲存最多 30 天之連線記錄，過期之連線記錄會自動被

刪除，為了讓連線記錄能被匯進使用者自己之專用資料庫，並做自己希望的加值利用(如 ISP 業者，可依用戶的使用流量來計費)，透過 Policy Log Export(PLExport) **程式，可在遠端隨時隨地進行連線記錄的加值利用。**PLExport **有適用於各種不同平台的版本。使用者可定義** PLExport **對於連線記錄輸出的純文字格式，再透過使用者自行撰寫的程式介面，將連線記錄匯入自己的資料庫，如圖** 2-17 **所示。**

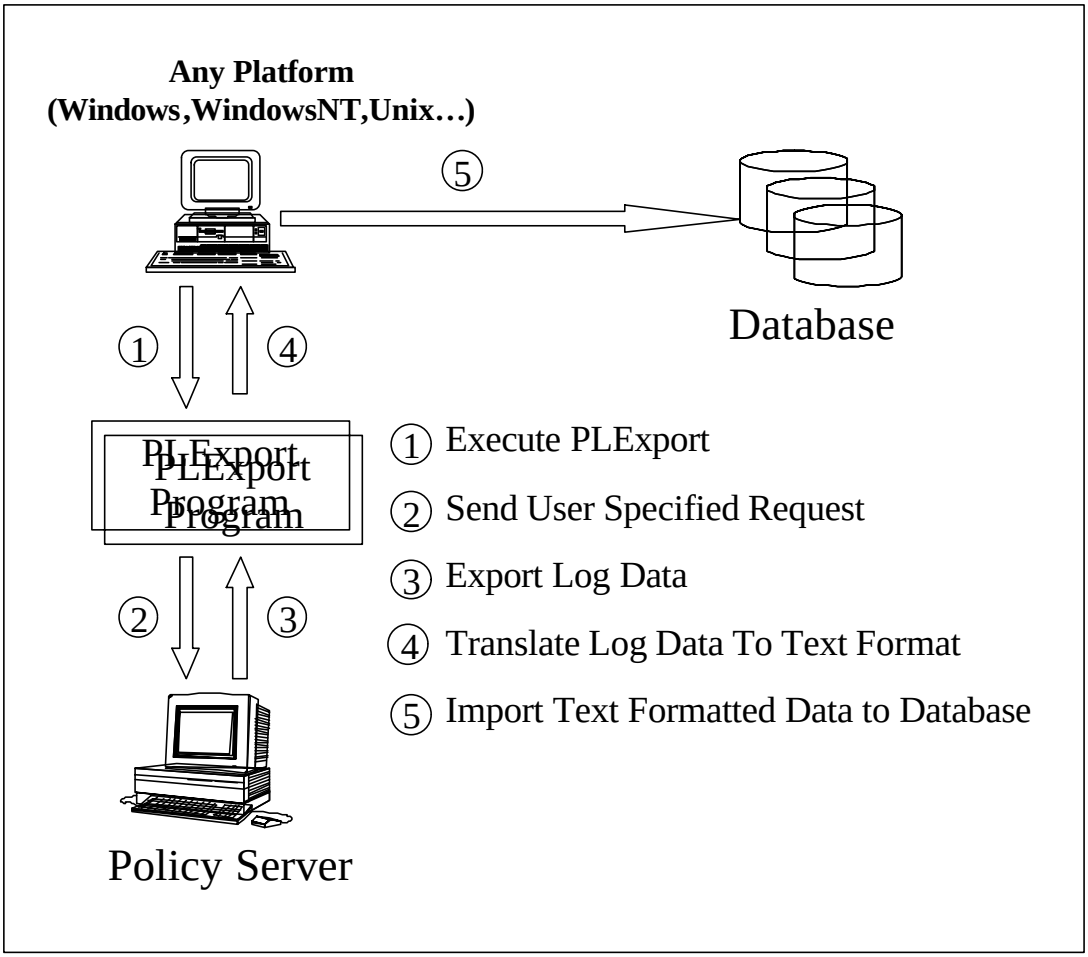


圖 2-17 **透過** PLExport **將資料再加值**

第3章 測試環境及運作流程

我們自己發展一套測式程式 –Catapult 來幫助測試，Catapult 是一支可扮 Client 或 Server 的程式，具有下列的特性：

- 可指定 IP/Port，以測試頻寬管理器能否正確的找到吻合之 Rule。
- 可指定收送的量，測試 Quota 的限制。
- 允許同時很多條連線的建立，交錯測試 Rule/Connection QoS。
- 接收端以圖形化介面，呈現出速度和時間的關係圖，可測試頻寬管理器有否達到指定的服務品質保證。

我們在頻寬管理器的 WAN 和 LAN 埠上，各接上一台 Switch[3]，再從 Switch 各接出兩台 PC，每一台 PC 皆裝上 Catapult，如圖 3-1 所示。

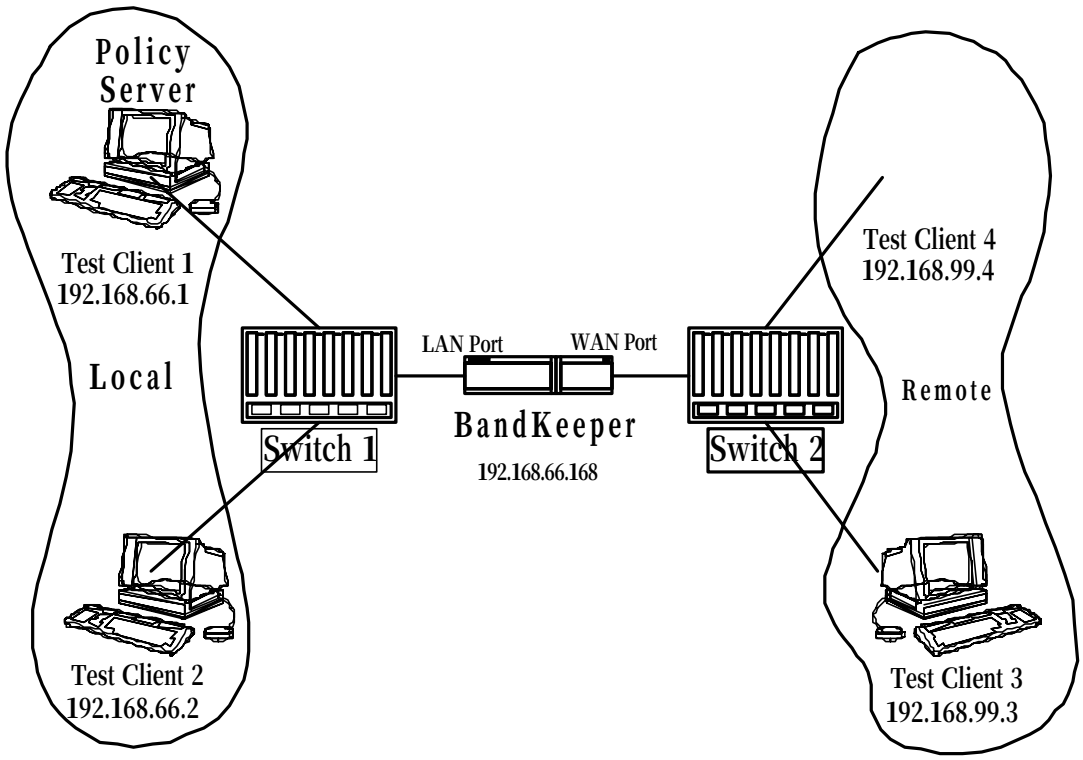


圖 3-1 政策性網路頻寬管理系統之測試環境

政策伺服器的位址是 192.168.66.1；頻寬管理器的位址為 192.168.66.168。為

[3]使用 Switch 的原因是為避免因為大量的測試資料彼此碰撞而達不到測試的效果。

了避免在測試期間，大量的連線記錄從頻寬管理器送往政策伺服器記錄而影響測試結果，我們先暫時將頻寬管理器傳送連線資料的時間間隔調至 30 分鐘。

在進入測試之前，我們先把 Address Book 定義好：

	Name	IP Address	Subnet Mask
Local	Policy Server (即 Test Client 1)	192.168.66.1	255.255.255.255
	Test Client 2	192.168.66.2	255.255.255.255
	Subnet 66	192.168.66.0	255.255.255.0
Remote	Test Client 3	192.168.99.3	255.255.255.255
	Test Client 4	192.168.99.4	255.255.255.255
	Subnet 99	192.168.99.0	255.255.255.0

以下我們就針對流量分類、服務品質保證、QoS 因時而變化和限定流量四種功能來測試：

測試一

目的：頻寬管理器能否正確依據政策規則做流量分類(flow classification)。

Rule 1: (Incoming)

Condition:

```
Source = Test Client 3
Destination = Test Client 2
Service = ANY
Schedule = ANY
```

Action:

```
No any limitation.
```

Rule 2: (Outgoing)

Condition:

```
Source = Subnet 66
Destination = Subnet 99
Service = FTP
Schedule = ANY
```

Action:

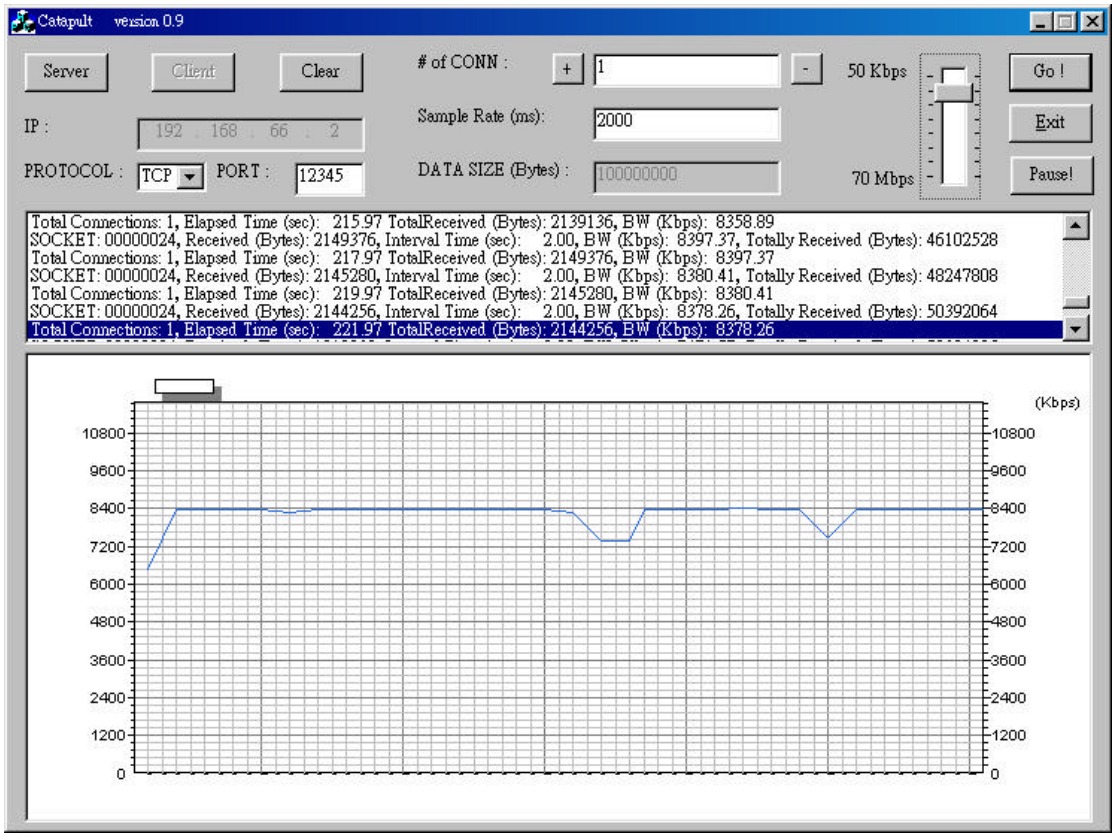
```
No any limitation.
```

步驟：

1. 使用 CuteFtp 從 Test Client 2 從 Test Client 3 做檔案傳輸服務時，連線可以

被建立，從 CuteFtp 可得知速度約有 930KB/s =7440Kb/s。(和 Rule 2 吻合)

2. 但從 Test Client 3 往 Test Client 1 沒有辦法建立起任何的連線。
3. 利用 Catapult 從 Test Client 3 往 Test Client 2 埠 12345 建立一條連線，從 Catapult 之接收端的時間-速度圖，也可看見將近有 8400Kb/s 之速度。(和 Rule 1 吻合)



結果：頻寬管理器有依照規則予以決定連線是否允許建立。

測試二:

目的：頻寬管理器能否達到政策規則之服務品質保證。

Rule 1: (Incoming)

Condition:

```
Source = Test Client 3
Destination = Test Client 2
Service = ANY
Schedule = ANY
```

Action:

```
Connection Maximum Rate (Remote➔Local) = 700Kb/s
```

Rule 2: (Outgoing)

Condition:

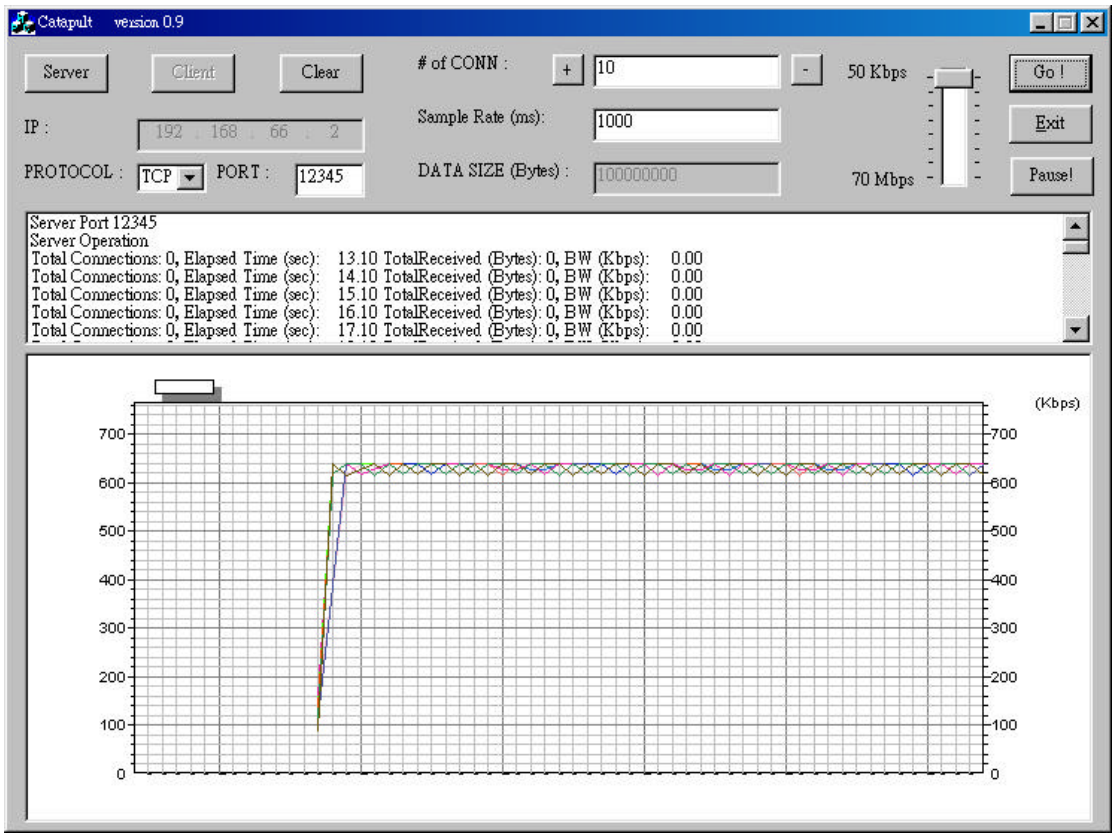
```
Source = Test Client 1
Destination = Test Client 4
Service = FTP
Schedule = ANY
```

Action:

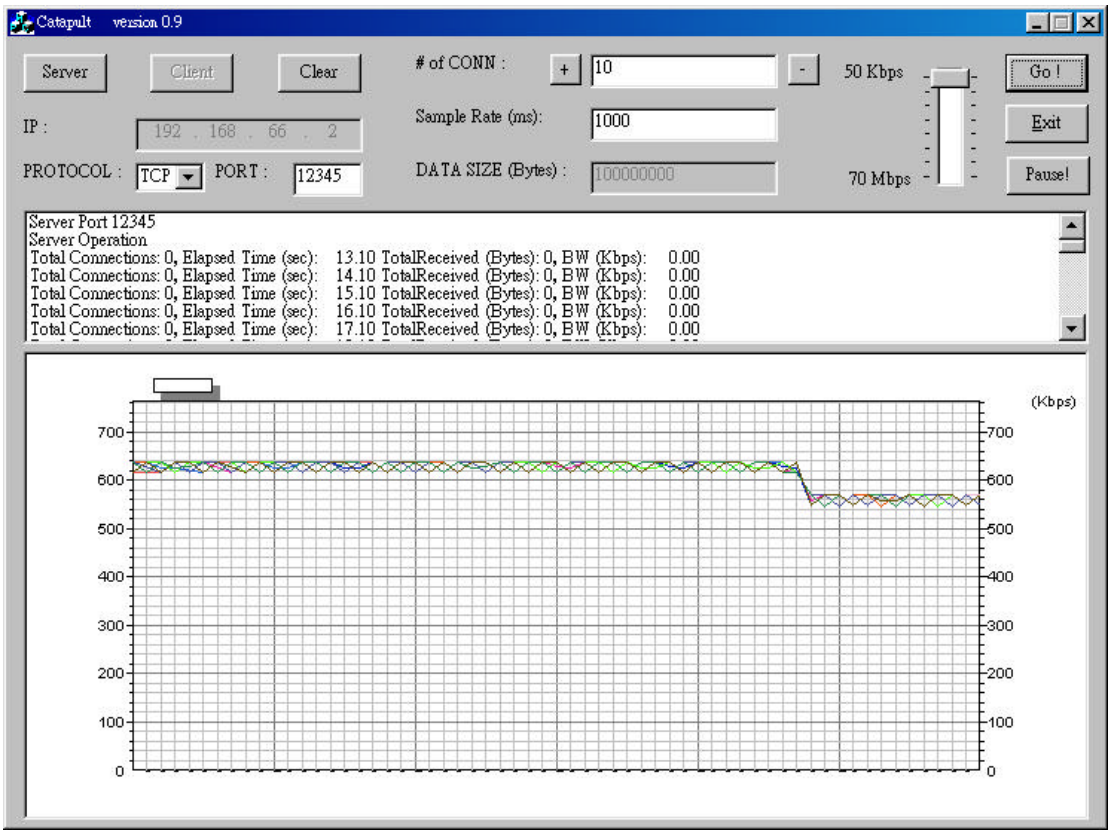
```
Rule Committed Rate (雙向) = 4000Kb/s
```

步驟：

1. **先利用** Catapult **從** Test Client 3 **向** Test Client 2 **埠** 12345 **建立** 10 **條連線，每條連線大約有** 630Kb/s **速度。(和** Rule 1 **吻合)**



2. **然後利用** CuteFtp **從** Test Client 1 **向** Test Client 4 **做檔案傳輸的協定。從** CuteFtp **中，很清楚地看見約有** 275KB/s = 2200Kb/s **的頻寬，這是因為** FTP **的協定中，有兩條的連線，一為控制連線，另一為資料連線，所以原來應有** 4000Kb/s **的保證頻寬，就被平分。(和** Rule 2 **吻合)**
3. **再從** Catapult **之接收端的時間-速度圖，也可看見每一條連線速度已經被限制至** 540Kb/s **的速度。**



4. **在結束傳檔之後，**Test Client 3 **至** Test Client 2 **的速度又會回到** 630Kb/s。

結果：頻寬管理器會依照政策規則予以服務品質的保證。

測試三:

目的：頻寬管理器是否具有 QoS 因時而變化的特性。

Rule 1: (Incoming)

Condition:

```
Source = Test Client 3
Destination = Test Client 2
Service = ANY
Schedule = 10:00 ~ 11:00
```

Action:

```
Connection Maximum Rate (Remote➔Local) = 300 Kb/s
```

Rule 2: (Incoming)

Condition:

```
Source = Test Client 3
Destination = Test Client 2
```

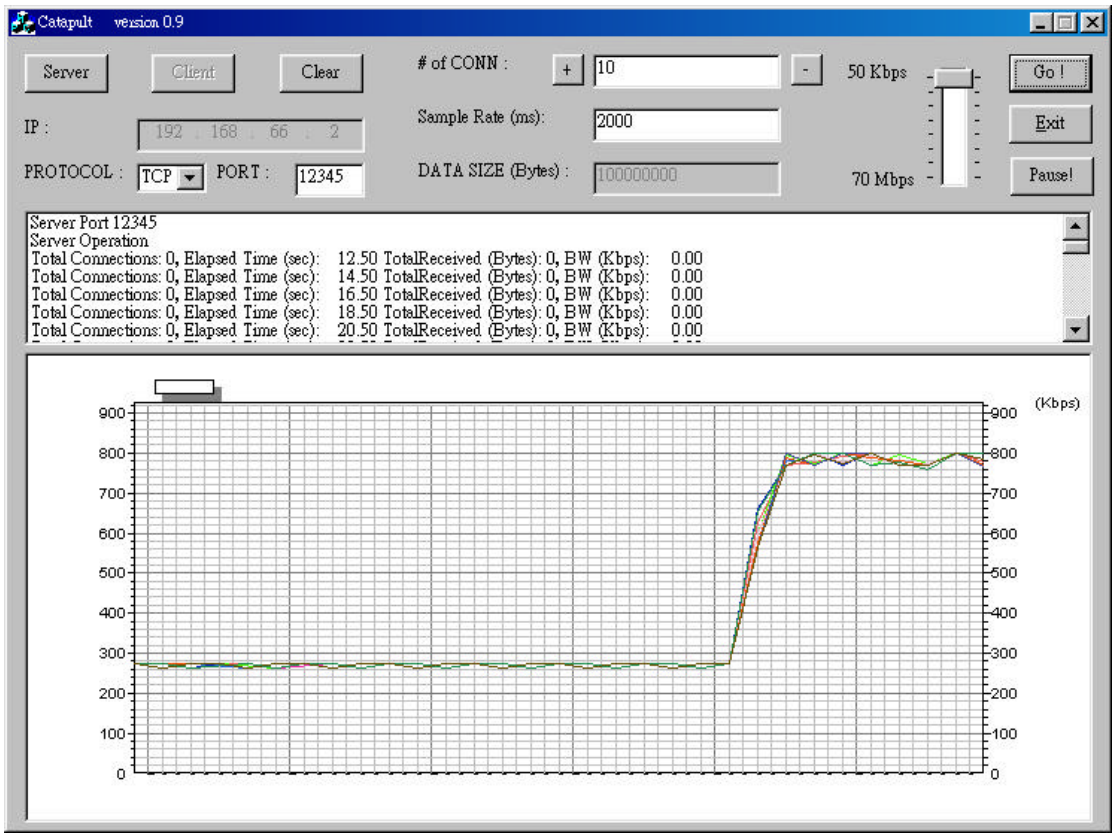
```
Service = ANY
Schedule = 11:00 ~ 12:00
```

Action:

```
Connection Maximum Rate (Remote➔Local) = 800 Kb/s
```

步驟：

1. **先將** BandKeeper **時間調至** 10:58 **分。**
2. **啟動** Catapult **從** Test Client 3 **向** Test Client 2 **埠** 12345 **建立** 10 **條連線，每條連線大約有** 270Kb/s **速度(和** Rule 1 **吻合)。在時間過** 11:00 **之後，從** Catapult **之接收端的時間-速度圖，也可看見每一條連線速度已經被提昇至** 780Kb/s **的速度。(和** Rule 2 **吻合)**



結果：頻寬管理器具有 QoS **因時而變化的特性。**

測試四:

目的：頻寬管理器能否達到限流量之要求(Quota)。

Rule 1: (Incoming)

Condition:

```
Source = ANY
Destination = ANY
Service = ANY
Schedule = ANY
```

Action:

```
No any limitation
```

Rule 2: (Outgoing)

Condition:

```
Source = ANY
Destination = ANY
Service = ANY
Schedule = ANY
```

Action:

```
No any limitation
```

步驟：

1. **在** Address Book **中的** Policy Server **項目中，把** Max Quoa **設為** 10MB。
2. **利用** CuteFtp **從** Test Client 1 **向** Test Client 4 **做檔案傳輸的協定，傳輸一個大於** 10M **位元組的檔案。在** CuteFtp **中，有已傳送的大小及比率，大約傳送到** 10M **時，連線就被停止，這是因** Test Client 1 **的流量超過指定的配額，除非過午夜才會重新計量，恢復連線。**

結果：頻寬管理器具有限制近端流量的功能。

除了上述這些功能外，尚有對每條連線限時(Duration)**、限量**(Max Octet)**等，也都已經成功的完成實作和測試。**

第4章 結語及未來發展

4-1. 結語

頻寬管理是目前多元化的網路使用環境下，管理網路資源的方式，網管人員依公司或組織的策略、應用軟體的特性和取向，來給予不同的頻寬。它的主要精神就是將頻寬使用在真正該使用的地方，以提昇網路使用效率。

但目前各家廠商所發展的頻寬管理系統使用不同的方法，也有著不同的特性。由於標準中，也僅僅提出政策性的網路系統架構，標準的政策規則描述法也正在發展，這使得各種頻寬管理系統沒有辦法彼此相容，造成政策伺服器只能對其可辨識的(device-aware)政策執行點集中控管。

4-2. 未來發展

整套的政策性網路頻寬管理系統已經完整的發展完畢，往後除了修正每個部份 Bug 之外，還有如下的相關工作：

4-2.1. 政策制定介面

現在已完成的政策制定介面，真的非常具親和力，但這所必需付出的代價就是冗長 Java Applet 下載時間的等待。這會對於遠端管理，會造成很大的致命傷。目前就是得找出一個可加快下載及執行速度的方法。再者，若當各種 Book 的定義項目個數達到數千以上，其執行速度也會降低，這或許要從資料結構上來解決。貫用了 C 語言中的指標(pointer)來建立快速的索引，來到 Java Applet 上，已不再支援指標，所以在 Java 上，如何建立快速的索引也是滿重要的課題。

4-2.2. 政策伺服器

政策伺服器應該要有能力拒絕服務某些非法的網路位址的連線而提供較佳的安全性。另外，政策制定介面、政策執行點與政策伺服器的通訊也能夠改良為安全通訊，PBMP 藉由使用 SSL 或 GSS 之類的網路安全協定，可以保証管理的內容不會被竊聽。

4-2.3. 頻寬管理器

目前的頻寬管理器是以軟體基底(software-based)的方法來實作，經過我們的測試，對於 10M 的網路，軟體的方法已足夠應付各種大小的封包而且幾乎可達等傳輸線速率(Wire-Speed)；對於 100M 的網路，若封包大小在 512 bytes 以上，還可達 70%的輸出率；但若對於超高速乙太網路而言，就只能使用硬體基底(hardware-based)的方法來實作。

4-3. 展望

我們利用政策性網路管理之架構實作的頻寬管理系統，精準地控制有限的頻寬，有效地利用每一分的頻寬資源，節省擴大頻寬的昂貴成本。對於那些非關鍵但常常造成網路頻寬浪費的應用程式，都可以被控制；而真正需要使用頻寬的應用，都可以得到該有的服務品質保證，讓公司或企業可從更重要的應用中得到更多的利益帶來更多的契機。

第5章 參考文獻

[1]. Hewlett-Packard Company, “A primer on Policy-based Network Management”, September, 24, 1999, Version 1.0.

[2]. M. Stevens, W. Weiss, H.Mahon, B.Moore, J.Strassner, G. Waters, A. Westerinen, J. Wheeler, IETF, Internet Draft, “Policy Framework”, September 13, 1999.

[3]. Hugh Mahon, Yoram Bernet, Shai Herzog, IETF, Internet Draft, “Requirements for a Policy Management System”, October, 22, 1999.

[4]. Francis Reichmeyer, Mark Stevens, IETF, Internet Draft, “A Unified Terminology for Policy Based Networking”, October 22, 1999.

[5]. Stardust.com Inc., “iBAND3 White Paper – Introduction to QoS Policies”, September 9, 1999.

[6]. Francis Reichmeyer, Kwok Ho Chan, David Durham, Raj Yavatkar, Silvano Gai, Keith McCloghrie, Shai Herzog, Andrew Smith, IETF, Internet Draft, “COPS Usage for Policy Provisioning”, February 1999.

[7]. Steve Waldbusser, Jon Saperia, Thippanna Hongal, IETF, Internet Draft, “Policy Based Management MIB”, March 8, 2000.

[8]. J. Strassner, E. Ellesson, B. Moore, Ryan Moats, Internet Draft, “Policy Framework LDAP Core Schema”, November 04, 1999.

[9]. J. Wroclawski, RFC 2211, “Specification for Controlled Load Network Element Service”, September, 1997.

[10].**黃能富著, 區域網路與高速網路 , 維科出版社,** 1998 **年** 6 **月增訂** 1 **版** 1 **刷**.