第十六章 SNMP 管理協定

16-1 SNMP 協定簡介

『簡易網路管理協定』(Simple Network Management Protocol, SNMP) 是在 1986 年由美國普渡大學的 Jeff Case 教授提出。當時設計 SNMP 的主要目的是希望能夠建立一套簡單的 TCP/IP 網路管理方式，直到目前為止，廣泛複雜的 Internet 網路，仍以 SNMP 為主要管理協定。許多廠商依照 SNMP 通訊協定製作出許多網路管理平台，例如 HP 公司的 Open-View、D-Link 公司的 D-View 等等。也就是說，任何網路設備製造廠商的網路管理系統，皆以 SNMP 通訊協定為基礎，才能使不同廠商之間的網路設備可以互相管理。目前 SNMP 有三個版本：SNMPv1 (RFC 1157)、SNMPv2 (RFC 1902) 與 SNMPv3 (RFC 2271 - 2275)，首先，我們介紹 SNMP 的基本原理，再分別介紹這些版本的未來發展。

16-2 SNMP 管理環境

在 SNMP 網路管理的環境之下，有三個主要元件：

▲ **SNMP 管理者 (SNMP Manager)：** 是 SNMP 的主要管理軟體，安裝於『網路管理系統』(Network Management System) 上，它負責向所管轄的管理設備索取管理訊息，或設定網路組態，為 SNMP 管理環境的主要控制設備。

▲ **SNMP 代理者 (SNMP Agent)：** 一般安裝於被管理的網路設備上，例如，路由器、主機電腦、橋接器等。SNMP Agent 收集網路設備上訊息，再以 SNMP 通訊協定和 SNMP Manager 通訊，達到管理的目的。

▲ **被管理物件 (Managed Object)：** 一般指被管理設備內之各種管理物件。例如，被管理之設備為橋接器，而其被管理物件為橋接器的有關設備，譬如，網路介面卡、緩衝器空間、過濾資料庫等等。被管理物件皆以**『管理訊息結構』**方式表達，且以樹狀識別碼 (Tree Identified) 方式，儲存於**『管理訊息資料庫』(Management Information Base, MIB)** 上。

SNMP 的網路管理架構，如圖 16-1 所示。網路上至少有一工作站（或兩個以上）安裝上『網路管理系統』(Network Management System, NMS)軟體，稱之為 SNMP Manager，例如，Open-View（HP）、D-View（D-Link），除了提供網路設備之圖形管理介面，並負責收集及管理各個網路設備。在 NMS 主機和被管理設備上都以 SNMP 通訊協定交換訊息，通訊協定堆疊關係如圖 16-2 所示。SNMP 通訊協定規範了管理者和被管理者之間的通訊方式，並提供認證與加密的功能。

基本上，SNMP 的管理模式式屬於『要求/回應』(Request/Response) 模式，SNMP Manager 下達命令給 SNMP Agent，SNMP Agent 再依照命令需求，收集有關網路的訊息，以『管理訊息結構』(Structure of Management Information) 格式回應給 SNMP Manager。SNMP Manager 收到訊息後，則可從事該網路設備的管理處理，例如，設定 IP 位址、啟動/停止連接埠口、更改路由表等處理。最簡單的應用是，由於一般 NMS 系統都有各種網路設備的虛擬面板，管理者不用到裝設地點，仍可以了解所有網路設備的運作情形，因此，SNMP 管理是 Internet 網路上不可或缺的設備。

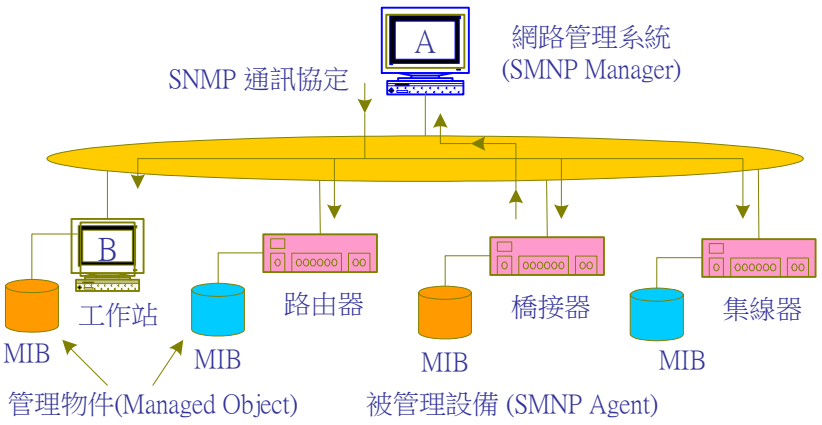


圖 16-1 SNMP 網路架構圖

16-3 SNMP 協定

16-3-1 SNMP 協定堆疊

圖 16-2 為 SNMP 網路管理的通訊協定堆疊，在傳輸層使用非連接方式的『使用者電報傳輸』(User Datagram Protocol, UDP)，連接到 UDP 161 埠口上，也就是說，SNMP Manager 和 SNMP Agent 之間是透過 UDP 161 埠口通訊。但 SNMP Manager 是利用 UDP 162 埠口來接收 Agent 所傳送過來的 Trap 訊息。資料表現層為管理物件資料的表示法，SNMP 使用標準化的『抽象語意表示法』(Abstract Syntax Notation - One, ASN-1) 來編碼，使不同廠商之間的網路設備可以互

相管理，亦是，在 SNMP Manager 和 SNMP Agent 之間所傳遞的訊息都以 ASN-1 格式包裝（請參考 16-4 節），才能達到異質電腦設備之間的網路管理。

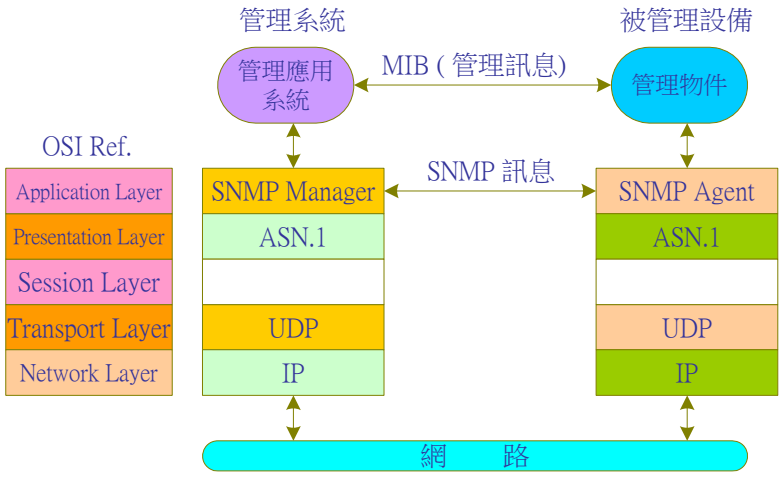


圖 16-2 SNMP 通訊協定之堆疊架構

16-3-2 SNMP 協定資料單元

圖 16-3 顯示 SNMPv1 與 SNMP v2 協定資料單元的格式，基本上，SMNP 訊息是透過 UDP 封包包裝，而以 IP 封包傳送，兩個版本在訊息標頭（SNMP Header）上都是一樣，但兩個版本的 SNMP PDU（Protocol Data Unit）會依照各種命令而不相同。一般在 SNMPv2 的環境裡無法直接使用 SNMPv1 的訊息格式，以及從事於管理工作，必須透過一個管理代理者的處理，這在 RFC 2089 中有詳細規範。以下說明各訊息欄位之功能：

(A) 訊息標頭（Message Header）

訊息標頭包含兩個欄位（圖 16-3 (a)）：

- **版本（Version）**：表示該 SNMP 封包的版本，0 表示 SNMPv1；1 表示 SNMPv2。
- **共同體（Community）**：表示某一管理群組織的共同體名稱，一般內定值為 Public。

(B) SNMPv1 一般協定資料單元

SNMPv1 的一般『**協定資料單元』(Protocol Data Unit, PDU)**包含有：Get、GetNext、Response 與 Set PDU 格式，它們的 PDU 格式都是相同，如圖 16-3 (b) 所示，各欄位功能如下：

▼ **PDU Type**：表示本 PDU 的命令型態：

0 → Get-Request

1 → Get-Next-Request

2 → Set Request

3 → Get-Response

▼ **Request ID**：當 SNMP Manager 下達某一命令，都會針對每一命令編碼一個 Request ID(隨機變數)，SNMP Agent 回應時，再針對哪一個 Request ID 回應訊息。

▼ **Error Status**：Set-Response PDU 存放發生錯誤訊息。如果 SNMP Manager 所下達的命令無法達成時，便依照錯誤狀態回應給 SNMP Manager。其它命令（Set、Get、GetNext）沒有使用此欄位。

▼ **Error Index**：Set-Response PDU 存放錯誤發生的索引位置，表示錯誤是發生在後面可變連結（Variable Bindings）上哪一位置上的管理物件。

▼ **管理物件**：每一管理物件以**《物件名稱，數值》(Object-n, Value-n)**來表示，可串接不定長度（依照 UDP 封包長度而定）。如果是 Get 與 GetNext 命令將不理會物件的數值（Value），如果是 Set 命令表示設定物件的數值，而 Response 表示回應物件的數值內容。

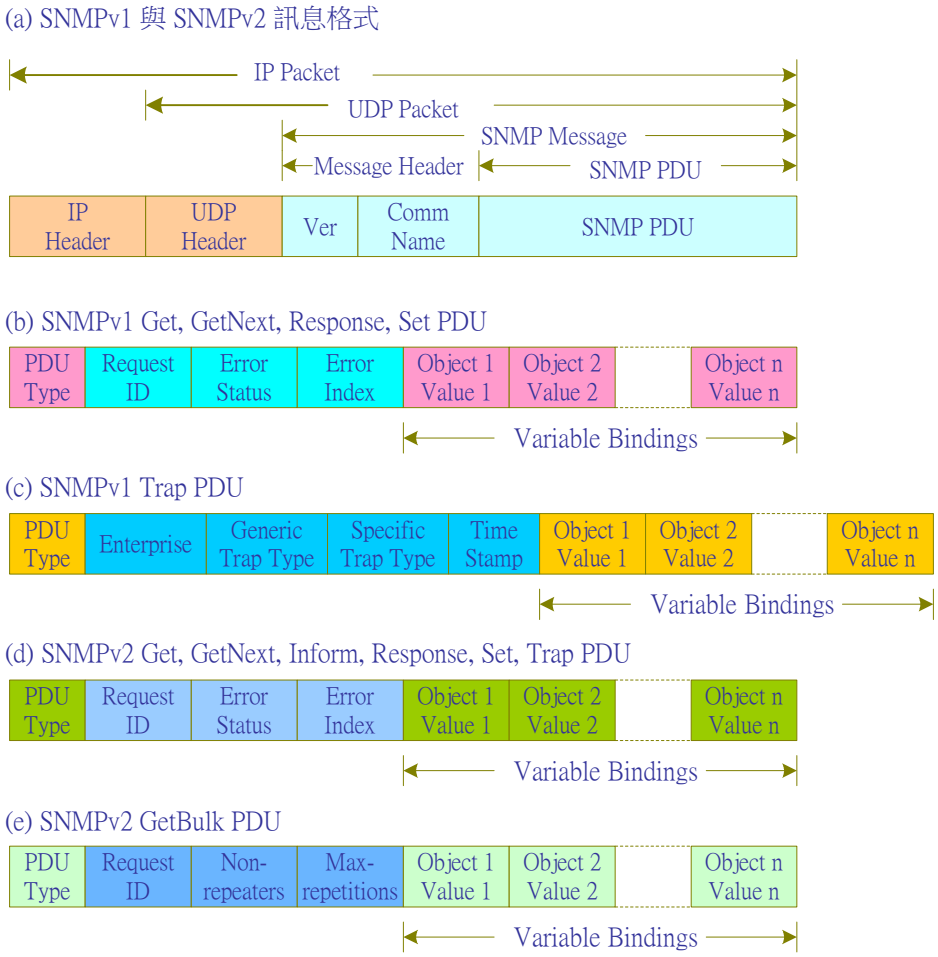


圖 16-3 SNMPv1 與 SNMPv2 訊息格式

(C) SNMPv1 Trap 協定資料單元

圖 16-3 (c) 顯示 SNMPv1 Trap PDU 格式，其中 PDU Type = 4，管理物件的表示也和一般 PDU 相同，其餘各欄位功能如下：

- ▼ **Enterprise**：標示所產生 Trap 的管理物件型態。
- ▼ **Agent Address**：提供 Trap 管理物件的 SNMP Agent 位址。
- ▼ **Generic Trap Type**：一般 Trap 型態。
- ▼ **Specific Trap Code**：在 Trap 型態中的一個特殊編碼。
- ▼ **Time Stamp**：表示所發生 Trap 的時間戳記。

(D) SNMPv2 一般協定資料單元

圖 16-3 (d) 顯示 SNMPv2 一般資料協定單元（PDU）的訊息格式，它可能是 Get、GetNext、Set、Response、Inform 或 Trap 命令，其中除了 PDU Type 欄位和 SNMPv1 不同外，其它欄位都相同，PDU Type 表示如下：

0 → Get-Request

1 → Get-Next-Request

2 → Set Request

3 → Get-Response

4 → 不使用（為了避開 SNMPv1）

5 → Get-Bulk-Request（不同的 PDU 格式）

6 → Inform-Request

7 → SNMPv2 Trap

(E) Get-Bulk-Request 協定資料單元

圖 16-3 (d) 為 GetBulk 的 PDU 格式，其中 PDU Type = 5，其它各欄位功能如下：

- **Request ID**：一個隨機變數的號碼，表示本訊息的編號，SNMP Agent 回應時，可標明依照哪一個要求回應訊訊息。
- **Non-repeaters**：描述本要求訊息中後面可變連結（Variable Binding）中物件事項（Object Instances）不要超過一次擷取資料的數量。
- **Max-repetitions**：定義最大可以重複擷取的次數，而這些管理物件未在 Non-repeater 上被註明不可重複擷取。

在 SNMPv1 協定只有定義五種錯誤訊息（Error Status），到了 SNMPv2 增加了 13 種錯誤訊息，主要是由 Response PDU 回應要求時，顯示管理訊息存取錯誤的狀態，我們由訊息名稱大略可以了解它的功能：

▲ **noError(0)**：正常動作，沒有錯誤。

▲ **tooBig(1)**：給變數的數值太長。

▲ **noSuchName(2)**：沒有此變數名稱。(與 Proxy 相容)

▲ **badValue(3)**：給變數數值錯誤。(與 Proxy 相容)

▲ **readOnly(4)**：該變數只能讀取不可寫入。(與 Proxy 相容)

▲ **GenErr(5)**：一般性的錯誤。

▲ **noAccess(6)**：表示該變數是不可存取的。

▲ **wrongLength(8)**：長度錯誤。

▲ **wrongEncoding(9)**：編碼錯誤。

▲ **wrongValue(9)**：給變數的數值錯誤。

▲ **noCreation(11)**：該變數是不可開啟的。

▲ **inconsistentValue(12)**：數值資料型態不一致性錯誤。

▲ **resourceUnavailabe(13)**：資源無效。

▲ **commitFailed(14)**：確認失敗。

▲ **undoFailed(15)**：重複執行失敗。

▲ **authorizationError(16)**：認證錯誤。

▲ **notWritable(17)**：該變數不可寫入。

▲ **inconsistentName(18)**：名稱不一制性的錯誤。

16-4 SNMP 運作程序

基本上，SNMP 是屬於『**要求/回應**』(**Request-Response**) 的管理模式，但在某些情況下，也允許觸發性 (Trap) 的傳遞資料。SNMP 的管理模式是，首先由 SNMP Manager 下達命令 (Set、Get Request)給 SNMP Agent，SNMP Agent 依照命令收集網路訊息後，再回應(Response)給 SNMP Manager (利用 UDP 161 埠口)。在正常情況之下，SNMP Agent 並不主動要求傳遞訊息給 SNMP Manager，除非有某些異常現象，再由 SNMP Agent 下觸發訊號 (Trap) 給 SNMP Manager (利用 UDP 162 傳輸埠口)，各種命令的運作情形如圖 16-4 所示。我們以在 SNMPv2 的運作程序來介

紹各個命令的運作情形，至於 SNMPv1 除了沒有 Get-Bulk 與 Inform 命令外，其餘運作程序也相同。

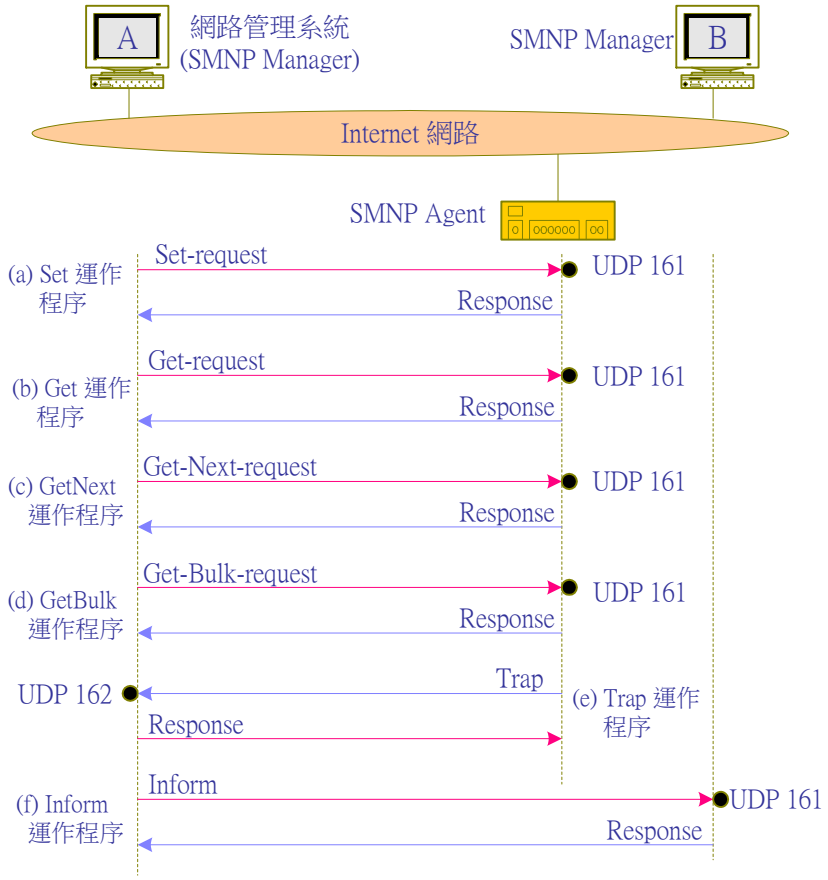


圖 16-4 SNMP 命令運作程序

(A) Get-Request 命令

SNMP Manager 發出要求 Get-Request 查詢網路設備的情況，SNMP Agent 收到訊息後，依照訊息內所詢問物件名稱（如圖 16-3 (b)）填入該物件的數值（Value）後，以 Get-Response 回應給 SNMP Manager。例如，SNMP Manager 要求詢問網路設備上 TCP 連線有幾條，則下達 Get-Request (tcpCurrEstab.0) 給 SNMP Agent。SNMP Agent 由設備上取得訊息後以 Get-Response(tcpCurrEstab.0 = 5) 給 SNMP Manager，表示目前已建立之連線有五條鏈路。如果 SNMP Agent 無法完成 Get-Request 命令時，則會以 Get-Response 回應錯誤訊息。

(B) Get-Response 命令

在 SNMP 通訊協定裡，SNMP Agent 除了 Trap 命令外都是以 Get-Response 命令回應所接收命令的處理結果給 SNMP Manager。如果 SNMP Manager 所下達命令無法達成時，SNMP Agent 會

將錯誤訊息填入 Get-Response PDU 內(如圖 16-3 (b))，以告知 SNMP Manager 發生錯誤的原因，並以 Error-index 指明擷取哪一個變數（或物件）發生錯誤。

(C) Get-Next-Request 命令

Get-Next-Request 和 Get-Request 命令非常類似，但在運作上有很大不同點。Get-Request 表示要索取某一管理物件的數值，一般來講這些管理物件都是以 MIB 的樹狀結構表示。Get-Next-Request 是使用在 MIB 樹狀結構不明確的情況下，探索某一管理物件的分支上的物件的數值。尤其在許多情況下，網路設備製造商會將管理物件依照自有的 MIB 結構排列(如 Private 物件)，它的樹狀結構也可能不是標準化，此情況下就可以利用 Get-Next-Request 來追蹤管理物件。SNMP Agent 也是以 Get-Response 來回應 SNMP-Manager 的查詢，但如果無法再追蹤下去時，則會回應該物件的值為 endOfMibView。

(D) Set-Request 命令

由 SNMP Manager 下達 Set-Request 命令，來更改 SNMP Agent 之網路設備的組態（Configuration），例如，該改 IP 位址，或啟動/停止某一連接埠口。如在執行上沒有問題，SNMP Agent 會以 Get-Response 回應原 Set-Request 的值。如果 Set-Request PDU 中所指定的物件無法設定改變時，會在 Get-Response PDU 的訊息欄位中指定錯誤原因（如 noAccess），並在索引欄位中指示出哪一個物件設定時發生錯誤。但一般廠商為了顧及安全性的問題，大多不提供 Set 命令功能。

(E) Trap 命令

Trap 是由 SNMP Agent 主動傳送訊息給 SNMP Manager，目的是通知已發生異常事件。在 SNMP 標準規格定義有七項異常狀況：冷啟動(coldStart)、溫啟動(warmStart)、介面斷線(linkDown)、介面恢復（linkUp）、驗證失敗（authenticationFailure）、企業專屬事件（enterpriseSpecific）。

(F) GetBulk-Request 命令（SNMPv2）

GetBulk-Request 是使用於一次要求大量資料。尤其，針對 MIB 的表格描述之物件，如使用 SNMPv1 方式，必須連續下幾次 Get-Next 才能取得整個表格的訊息，而使用 GetBulk 一次就能取得整個表格的訊息。如果 SNMP Agent 無法回應整個系列訊息，可以只回應某些訊息（Get-Response）。

(G) Inform-Request 命令（SNMPv2）

SNMPv2 允許同一個網路下有多個網路管理系統，每一網路管理系統上都有一個 SNMP Manager。這些 SNMP Manager 之間就用 Inform-Request 來互相傳遞訊息，當對方 SNMP Manager 接收訊息時，還是以 Get-Response 回應。

16-5 SNMP 共同體

如果我們從圖 16-4 來觀察，該管理設備能接受工作站 A 的 SNMP 命令，同樣的它也可以接受工作站 B 的命令，也就是說，某一個 SNMP Agent 可能會接受多個 SNMP Manager 管理控制。又在許多實務上，一個 SNMP Manager 確有需要管理多個網路設備，這牽涉到一個問題，一個 SNMP Agent 到底是要受哪一個 SNMP Manager 管理的問題。因此，SNMP 協定就制定一個**『共同體』(Community)**的管理辦法，我們可以將某一群組的網路設備和管理工作站設定一個共同體，並給予一個共同體名稱，傳送訊息時便將這共同體名稱放置於訊息標頭內，如圖 16-3 (a) 的 Community 欄位裡。Manager 和 Agent 之間就利用共同體名稱來辨識是否接收訊息，更進一步，我們也可以將一些認證的訊息放置在這個欄位，從事認證的工作（或密碼確認）。

共同體標頭雖然可以解決管理者和被管者之間的分辨功能，但在實際應用上還是有許多問題，譬如遇到入侵者使用重複攻擊法，便能輕易地癱瘓整個網路系統。因此許多廠商將該欄位當作密碼確認使用，並取消會影響網路設定的 Set 命令取消，以減少被攻擊的可能，因此，對整個 SNMP 協定而言，僅能監看網路狀態而言，大大限制了 SNMP 協定（SNMPv1 與 SNMPv2）在網路上的應用。

16-6 管理訊息結構

我們由圖 16-3 (b) ~ (e) 上可觀察到一個可變連結的管理物件及其數值，這些管理物件如何來定名和數值如何表示，這就是**『管理訊息結構』(Structure of Management Information, SMI)**表示法(RFC 1442)。SNMP 將每一管理物件皆以**『管理訊息庫』(Management Information Base, MIB)**編排，而給於一個**『物件識別值』(Object Identify)**，並且依照物件性質都會有一個內容數值。在 SNMP 通訊協定上，無論物件識別值或該物件的內容都以 ASN.1 標準資料格式傳送，才能達到異質網路設備之間的管理，因此，針對管理訊息結構有下列資料型態：

- **Integer**：表示精確長度沒有限制的整數資料型態。例如，介面 MTU 的數值。

- **Octet String**：表示零個或更多個八位元的位元組所組成的字串，每一個位元組都有一個 0 到 255 之間的值。
- **NULL**：表示該變數沒有值，如 Get 或 GetNext 請求，所有物件的值都以 Null 表示，因為這個值是使用於查詢，而不是設定。
- **Object Identifier**：代表授權命名物件的資料型態，物件包含描述 MIB 樹狀結構的數值串列，如，1.3.6.1.2.1.6.13.1 表示 tcpConnTable 物件的識別值。
- **Sequence（Sequence Of）**：用來表示一個排序的串列，裡面包含零到多個元件，這些元件可能是其它資料型態或 Sequence 型態的資料(與 C 語言的 Structure 很類似)。Sequence Of 包含一個排序的串列，而串列中的每一個元素都擁有相同的資料型態。
- **Counter（Counter 32）**：代表一個非負數的計數器，可一直增加到最高數值（232-1）再回復零開始。但 Counter 沒有定義起始值。
- **Counter 64**：如同 Counter 32，但最高數值為 264-1。
- **Display String**：表示 0 到多個位元組所組成的字串，但每一個位元組都必須是從 NVT ACSII 集合中而來的一個字元，在 MIB-II 中所有此型態的變數都必須在 255 個字元以內，但長度可以為 0。
- **Gauge（Gauge 32）**：表示範圍由 0 到 232-1 的非負數整數，它的值可以增加或減少，但會鎖定在最大值。也就是說，如果一個值增加到 232-1，它便會保持該值，一直到被重新設定為止。譬如，tcpCurrEstab 的 ESTABLISHED 或 CLOSE_WAIT 狀態的連線號碼。
- **IPAddress**：長度為 4 的 Octet String，用於表示 IP 位址。
- **PhysAddreee**：表示網路實體位址的資料型態，譬如 Ethernet 網路是以長度為 6 的 Octet String 表示。
- **Opaque**：可以提供通過使用 Octet String 資料型態中任意資料的能力。
- **TimeTicks**：也是一個非負數的整數資料型態，表示一個從某個時間開始算起，以百分之一秒為單位的計數器。不同變數可以從不同的時間指定計數器，當變數在 MIB 裡宣告時，

就會用於指定用於每個變數的時期，譬如，sysUpTime 變數代表 Agent 已經啟動花費的百分之一秒的數目。

16-7 MIB 訊息資料庫

『管理訊息資料庫』(Management Information Base, MIB) 是描述被管理物件的資料結構。SNMP 被設計成可以管理各種網路設備，造成它所管理的訊息也隨著設備有成千上萬種，而每種網路或設備對自己資料的表達方式也不會相同，為了讓這些資訊納入同一套管理系統，就必須採用一套抽象的語法，來描述所有類型的訊息，因此，SNMP 定義了**『管理訊息資料庫』(MIB)**，以階層式來描述所有被管理物件的屬性，並稱這些被管理訊息為**『SNMP 物件』(SNMP Object)**。

16-7-1 管理物件表示法

SNMP 針對每一個 SNMP Object 都以**『物件描述樹』(Object Identified Tree)**來描述。因此，MIB 為一樹狀結構 (MIB Tree)，它的每一節點代表一個管理群組 (如 internet、private、snmp) 或管理物件(如 system、ip、tcp)，而其末端(leaf)為 SNMP Object 的訊息(如 tcpRtoAlgorithm)。每一節點或端點都有編號，此邊號稱之為**『物件識別值』(Object Identifier, OID)**。SNMP 的物件描述樹如圖 16-5 所示。

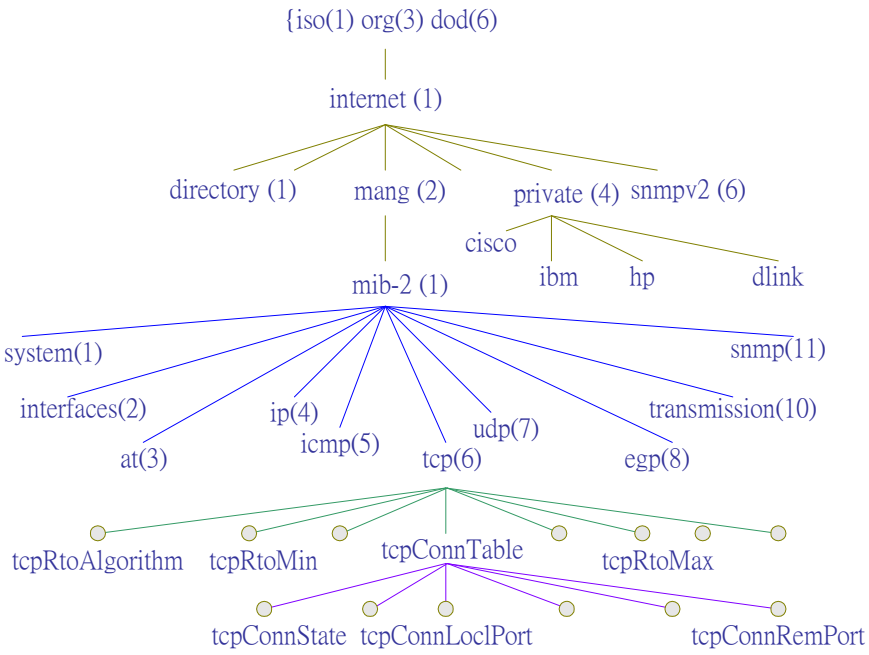


圖 16-5 MIB 樹狀結構部份樣本

我們用 udp 群組來說明管理物件的內容數值、MIB 編排的順序、以及這些功能的簡單例子。如圖 16-6 中 udp 群組包含了 4 個管理物件（udpInDatagrams、udpNoPort、udpInErrors 與 udpOutDatagrams）和一個表格（udpTable），在樹狀結構圖中以『●』表示某一樹狀描述的分支端點，它並沒有真正的數值內容，因此無法讀取或寫入。另外以『■』表示描述樹的分葉，它代表著某一管理物件，也表示存放著一個數值內容可以讀取，在某些情況下允許更改該數值內容。

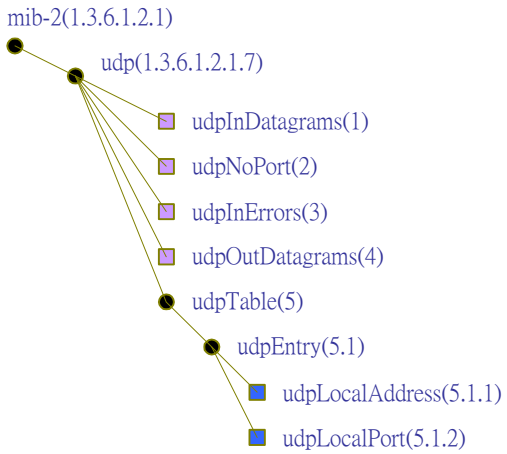


圖 16-6 udp 群組的樹狀結構

我們用表 16-1 來登錄這些管理物件，並且編排某一物件的號碼，以 udp 群組物件編號是 1.3.6.1.2.1.7，則 udpInDatagrams(1) 為 1.3.6.1.2.1.7.1，或者以 udp.1 表示之；資料型態欄位表示該數值變數的型態，可能是 Integer32、Counter32、或其它型態；讀/寫欄位表示該數值是否允許變更，如標示『NA』表示不可以讀取（該物件為分支端點）；描述欄位是對該管理物件功能作簡單的描述。

表 16-1 udp 群組的管理物件（1.3.6.1.2.1.7）

物件名稱	資料型態	讀/寫	描　　述
udpInDatagrams(1)	Counter		傳入的 UDP 封包總數
udpNoPorts(2)	Counter		沒有特定連接埠口的 UDP 封包總數。
udpInErrors(3)	Counter		傳入的錯誤 UDP 封包總數。
udpOutDatagrams(4)	Counter		傳出 UDP 封包總數。
udpTable(5)	Sequece of	NA	Udp 表格。

在圖 16-6 中有一個 udpTable 表格（UDP 傾聽表格），資料型態為 Sequence of udpEntry，它的結構非常類似 C 語言的 Structure 資料型態。udpEntry 描述表格內容如下：

```
udpEntry ::= Sequence {
```

```
        udpLocalAddress     IpAddress,
        udpLocalPort        Integer,
    }
```

表 16-2 為 udpTable 表格的物件編排號碼及其內容描述，其中 udpLocalAddress 的物件識別標號為 1.3.6.1.2.7.5.1.1，udpLocalPort 是 1.3.6.1.2.7.5.1.2。如以一般表示方法為 <udpLocalAddress>.<udpLocalAddress>，簡單的說，就是哪一個本地網路位址下的哪一個傳輸埠口正在傾聽等待接收資料。

表 16-2 udpTable 表格之管理物件（1.3.6.1.2.7.5）

物件名稱	資料型態	讀/寫	描　　述
udpEntry(1)	Sequence	NA	Udp 表格入口。
udpLocalAddress(1.1)	IpAddress		本地 IP 位址。
udpLocalPort(1.2)	Integer		本端埠口。

16-7-2 引例識別

當 SNMP 參照到 MIB 裡的每一變數時，這些變數必須被辨識出來，以供取得或設定其內容，這裡面有兩個重點：第一，只有葉部的節點可以被參考；第二，SNMP 並不操作表格內的全部列數或全部行數的內容。由圖 16-5 中可以看出，SNMP 不能直接讀取 mib-2、udp、udpTable 和 udpEntry 的內容，因為它們不是描述樹中的葉部。但我們在追蹤描述樹時，可能會由一個端點往下一個端點來探索，針對每一端點如何引導下一個節點（或端點），稱之為『引例識別』(Instance Identified)，有下列兩種方式：

(A) 簡單變數

簡單變數表示在描述樹中的葉部，它只描述某一個管理物件，也無法再由這個分葉再往下探索，因此都是藉由將『.0』附加到物件描述值的後面。例如，1.3.6.1.2.7.1.0（udpIndatagram.0） 表示指定到 udpIndatagram 變數的內容。簡單變數在存取方面比較簡單，直接指定到該變數位址即可，但是執行 SNMP 命令時，只能使用絕對位址，而不能使用相對位址。如 SNMP Manager 以標準命令格式擷取變數內如如下：

GetRequest(iso.org.dod.internet.mgnt.mib-2.udp.udpInDatagrams.0) 或

GetRequest(1.3.6.1.2.7.1.0)

則 SNMP Agent 回應：

GetResponse(iso.org.dod.internet.mgnt.mib-2.udp.udpInDatagrams.0=30) 或

GetResponse(1.3.6.1.2.7.1.0=30)

有些文件（如 RFC）為了簡化書寫繁雜，將其簡化成 GetRequest (udpInDatagrams)或 GetResponse(udpInDatagrams=30)，但它也是表示絕對位址的意思。

我們可以利用 snmputil 命令來擷取變數的內容（snmputil 是 Windows 2000 的 SNMP Manager 程式，請參考附錄安裝手冊），snmputil 命令格式為：

C:\>snmputil [get|getnext|walk] agent community oid [oid ….]

其中：get|getnext|walk 為擷取方式，get 表示擷取某一 oid（Object Identified）變數內容，因此必須指定某一絕對位址，而 oid 後面必需加上『.0』; getnext 表示 oid 下一變數的內容；walk 表示按照 MIB-II 描述樹追蹤下去。agent 是 SNMP Agent 的識別碼，一般皆以 IP 位址表示。community 為共同體的識別字，除非 Agent 上有特別設定，否則都以內定值是 public 表示。oid 是物件識別碼，標準表示方式為『.3.6.1.2.1.7.1』，但 snmputil 允許以較簡潔的『udp.1』表示。

```
C:\Resource Kit>snmputil get 163.15.2.33 public udp.1.0    <Enter>
Variable = udp.InDatagrams.0
Value = Counter32 915
C:\Resource Kit>snmputil getnext 163.15.2.33 public udp.1.0    <Enter>
Variable = udp.udpNoPorts.0
Value    = Counter32 7
```

上述第一個 snmputil 是以 get 命令擷取，而指定到 oid 位址是 udp.1.0，第二命令則以 getnext 命令擷取，雖然指定到同一個變數，但它是擷取下一個變數（udpNoPorts.0）的內容。

(B) 表格

圖 16-7 為 UTP 傾聽表格（udpTable）與 MIB 描述樹之間的關係。udpTable 中有兩個屬性（Attribute）：udpLocalAddress 和 udpLocalPort 變數分別在描述樹的分葉上，如用一般表格表示方式，表格中的『行』表示各變數的內容，也就是描述樹的分葉（如 udpLocalAddress）；而『列』是哪一筆資料，如第一筆資料是 0.0.0.0:7，表示本地網路位址是 0.0.0.0，埠口位址是 7，正在傾聽是否有資料傳送過來，另外四筆資料也是一樣（只節錄某些資料）。

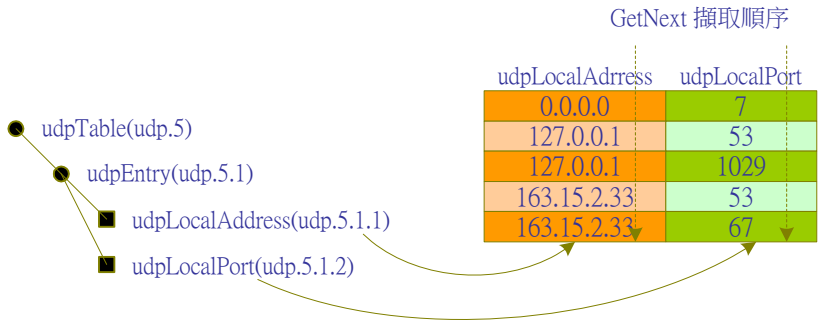


圖 16-7 udpTable 表格與描述樹關係的範例

MIB 是以物件識別值（Object Identifier, OID）的順序排列，排列方式有一個特殊的隱含性，也就是 MIB 的所有紀錄都是以它們物件識別值為順序，作為辭典編纂順序。如以圖 16-7 中的 udpTable 表格內容為例，每一列（或稱每一紀錄）的物件識別值如表 16-3 所示，但 MIB 字典編纂是以每一行（或稱每一物件確認值）為順序排列，如同表 16-3 的紀錄順序經過以物件識別值為順序排例，其排列結果如表 16-4 所示。

表 16-3 udpTable 表格內每一列物件確認值

列	物件確認值（OID）	簡寫名稱	值（Value）
1	1.3.6.1.2.7.5.1.0.0.0.0.7 1.3.6.1.2.7.5.2.0.0.0.0.7	udp.udpLocalAddress.0.0.0.0.7 udp.udpLocalPort.0.0.0.0.7	0.0.0.0 7
2	1.3.6.1.2.7.5.1.127.0.0.1.53 1.3.6.1.2.7.5.2.127.0.0.1.53	udp.udpLocalAddress.127.0.0.1.53 udp.udpLocalPort.127.0.0.1.53	127.0.0.1 53
3	1.3.6.1.2.7.5.1.127.0.0.1.1029 1.3.6.1.2.7.5.2.127.0.0.1.1029	udp.udpLocalAddress.127.0.0.1.1029 udp.udpLocalPort.127.0.0.1.1029	127.0.0.1 1029
4	1.3.6.1.2.7.5.1.163.15.2.33.53 1.3.6.1.2.7.5.2.163.15.2.33.53	udp.udpLocalAddress.163.15.2.33.53 udp.udpLocalPort.163.15.2.33.53	163.15.2.33 53
5	1.3.6.1.2.7.5.1.163.15.2.33.67 1.3.6.1.2.7.5.2.163.15.2.33.67	udp.udpLocalAddress.163.15.2.33.67 udp.udpLocalPort.163.15.2.33.67	163.15.2.33 67

表 16-4 udpTable 表格之 MIB 辭典編纂順序

行	物件確認值（OID）	簡寫名稱	值（Value）
1	1.3.6.1.2.7.5.1.0.0.0.0.7 1.3.6.1.2.7.5.1.127.0.0.1.53 1.3.6.1.2.7.5.1.127.0.0.1.1029 1.3.6.1.2.7.5.1.163.15.2.33.53 1.3.6.1.2.7.5.1.163.15.2.33.67	udpLocalAddress.0.0.0.0.7 udpLocalAddress.127.0.0.1.53 udpLocalAddress.127.0.0.1.1029 udpLocalAddress.163.15.2.33.53 udpLocalAddress.163.15.2.33.67	0.0.0.0 127.0.0.1 127.0.0.1 163.15.2.33 163.15.2.33
2	1.3.6.1.2.7.5.2.0.0.0.0.7 1.3.6.1.2.7.5.2.127.0.0.1.53 1.3.6.1.2.7.5.2.127.0.0.1.1029 1.3.6.1.2.7.5.2.163.15.2.33.53 1.3.6.1.2.7.5.2.163.15.2.33.67	udpLocalPort.0.0.0.0.7 udpLocalPort.127.0.0.1.53 udpLocalPort.127.0.0.1.1029 udpLocalPort.163.15.2.33.53 udpLocalPort.163.15.2.33.67	7 53 1029 53 67

我們以 snmputil 的 GetNext 命令格式來擷取 utpTable 的表格內容，也可容易的發現，GetNext 所追蹤的路徑就如圖 16-7 中的虛線方向，而描述樹中的 udpLocalAdress（udp.5.1.1）就是 MIB 辭典中的第一行，而 udpLocalPort（udp.5.1.2）就是辭典中的第二行。我們依照圖 16-7 擷取 udpLocalAddress（udp.5.1.1）所追蹤的範例如下：

```
C:>snmputil getnext 163.15.2.33 public udp.5.1.1    <Enter>
Variable = udp.udpTable.udpEntry.udpLocalAddress.0.0.0.0.7
Value    = IpAddress 0.0.0.0
C:>snmputil getnext 163.15.2.33 public udp.5.1.1.0.0.0.7    <Enter>
Variable = udp.udpTable.udpEntry.udpLocalAdrress.127.0.0.1.53
Value    = IpAddress 127.0.0.1
C:>snmputil getnext 163.15.2.33 public udp.5.1.1.127.0.0.1.53    <Enter>
Variable = udp.udpTable.udpEntry.udpLocalAdrress.127.0.0.1.1029
Value    = IpAddress 127.0.0.1
C:>snmputil getnext 163.15.2.33 public udp.5.1.1.127.0.0.1.1029    <Enter>
Variable = udp.udpTable.udpEntry.udpLocalAdrress.163.15.2.33.53
Value    = IpAddress 163.15.2.33
C:>snmputil getnext 163.15.2.33 public udp.5.1.1.163.15.2.33    <Enter>
Variable = udp.udpTable.udpEntry.udpLocalAdrress.163.15.2.33.67
Value    = IpAddress 163.15.2.33
```

同樣的，我們擷取 udpLocalPort（udp.5.1.2）的追蹤的路徑如下：

```
C:>snmputil getnext 163.15.2.33 public udp.5.1.2    <Enter>
Variable = udp.udpTable.udpEntry.udpLocalPort.0.0.0.0.7
Value    = Integer32 7
C:>snmputil getnext 163.15.2.33 public udp.5.1.2.0.0.0.0.7    <Enter>
Variable = udp.udpTable.udpEntry.udpLocalPort.127.0.0.1.53
Value    = Integer32 53
C:>snmputil getnext 163.15.2.33 public udp.5.1.2.127.0.0.1.53    <Enter>
Variable = udp.udpTable.udpEntry.udpLocalPort.127.0.0.1.1029
Value    = Integer32 1029
C:>snmputil getnext 163.15.2.33 public udp.5.1.2.127.0.0.1.1029    <Enter>
Variable = udp.udpTable.udpEntry.udpLocalPort.163.15.2.33.53
Value    = Integer32 53
C:>snmputil getnext 163.15.2.33 public udp.5.1.2.163.15.2.33.53    <Enter>
Variable = udp.udpTable.udpEntry.udpLocalPort.163.15.2.33.67
Value    = Integer32 67
```

16-8 MIB 管理物件群組

IETF 於 1989 年首次提出 MIB-I 標準，又在次年提出 MIB-II 標準。MIB-I 有八個管理群組，共有 114 個管理物件；而 MIB-II 修正後共有十個管理群組，171 個管理物件，如表 16-5 所示。目前還有許多管理物件，漸漸加入 MIB-II 標準內，例如，電源管理裝置、以及其他網路設備（請參考相關的 RFC）。

表 16-5 MIB-I 與 MIB-II 之管理群組

管理群組	MIB-I	MIB-II	功能說明
system	3	12	系統相關資訊（如，名稱、啟動時間）
interface	22	23	網路介面之相關訊息
at	3	3	網路位址轉換（如，IP/MAC 之轉換）
ip	33	38	IP 通訊軟體之相關訊息
icmp	26	26	ICMP 通訊軟體之相關訊息
tcp	17	19	TCP 通訊軟體之相關訊息
udp	4	6	UDP 通訊軟體之相關訊息
egp	6	18	外部閘門之路徑協定的相關訊息
transmission		0	傳輸媒介之相關訊息
snmp		30	SNMP 通訊軟體之相關訊息
總 計	114	171	

每個管理物件的內容都依照**『管理訊息結構』**來宣告，該內容數值的存取方式也區分為 ReadOnly 或 Read/Write。以下說明各種管理模組的管理物件。

16-8-1 系統群組

系統群組（System）提供 SNMP Agent 執行時有關系統或是裝置的資訊，這些資訊也都以一個物件識別值表示。系統群組的物件識別碼為 1.3.6.1.2.1（如表 16-6 所示），而它所屬的管理物件再由 System 端點分支下來，如 sysDescr (1) 的識別碼為 1.3.6.1.2.1.1。在每一個分支的管理物

件也可能再有其它分支物件，這已不在我們的討論範圍，表 16-6 僅列出系統群組下的管理物件，如要更詳細的說明請參考 RFC-1450。

表 16-6 system 群組之管理物件（1.3.6.1.2.1）

物件名稱	資料型態	讀/寫	描　　述
sysDescr(1)	DisplayString		裝置的描述。
sysObjectID(2)	ObjectID		裝置製造商的識別 ID。
sysUpTime(3)	TimeTicks		該裝置再被啟動後的時間計數。
sysContact(4)	DisplayString	是	該裝置的聯絡人訊息。
sysName(5)	DisplayString	是	裝置名稱或完整網域名稱。
sysLocation(6)	DisplayString	是	裝置安裝的實際位置。
sysServices(7)	Integer		裝置提供的服務。
sysOrLastChange(8)	TimeTicks		任何物件上一次發生變動的時間計數。
sysORTable(9)	Sequence of	NA	代理物件資源的動態設定表。

system 群組中包含了一個表格 sysORTable（1.3.6.1.2.1.9），它的作用是當做被管理物件扮演 SNMPv2 代理功能時，其所支援管理之物件資源的動態設定表（SNMPv1 無此功能）。sysORTable 表格內容如表 16-7 所示。

表 16-7 sysORTable 表格之管理物件（1.3.6.1.2.1.9）

物件名稱	資料型態	讀/寫	描　　述
sysOREntry(1)	Sequence	NA	表格特定入口。
sysORIndex(1.1)	ObjectID		使用指標在 sysORTable 上。
sysORID(1.2)	Display String		物件標號，和 sysObjectID 相似。
sysORDescr(1.3)	Display String		物件描述。
sysORUpTime(1.4)	TimeTicks		物件重新啟動時間計數。

16-8-2 介面群組

介面群組（Interface）提供安裝在裝置內的網路介面訊息及設定。這群組有一個 ifNumber 物件是用來描述已安裝在系統上的網路介面總數，無論是否已啟動使用。另一個物件 ifTable 是一個表格，其中每一列代表每一個介面，每個介面是以 ifIndex 物件來作索引，並且包含從 1 到 ifNumber 物件內的數值。表 16-8 為介面群組的管理物件，而表 16-9 為 ifTable 內的管理物件，其中包含下列三大類：

(1) 介面訊息（Interface Information）：由 ifEntry.1 ~ ifEntry.9。

(2) 進來交通訊息（Incoming Traffic）：由 ifEntry.10 ~ ifEntry.15。

(3) 外出交通訊息（Outgoing Traffic）：由 ifEntry.16 ~ ifEntry.22。

表 16-8 interface 群組之物件（1.3.6.1.2.2）

物件名稱	資料型態	讀/寫	描　　述
ifNumber(1)	Integer		該裝置內網路介面總數。
ifTable(2)	Sequence of	NA	介面表格中介面資料列表。

表 16-9 ifTable 表格之管理物件（1.3.6.1.2.2.2）

物件名稱	資料型態	讀/寫	描　　述
ifEntry(1)	Sequence	NA	表格特定入口。
ifIndex(1.1)	Integer		一個 MIB 參考定義。
ifDescr(1.2)	Display String		介面描述，如 eth0、ppp0。
ifType(1.3)	Display String		介面的類型。
ifMtu(1.4)	Integer		介面的最大傳輸單元。
ifSpeed(1.5)	Gauge		資料傳輸速率。
ifPhysAddress(1.6)	physAddress		介面實體位址（Ethernet 位址）。
ifAdminStatus(1.7)	Integer	是	控制介面 1=up、2=down、3=test。
ifOperStatus(1.8)	Integer		介面狀態 1=up、2=down、3=test。
ifLastChange(1.9)	TimeTicks		最近更新運作的時間計數。
ifInOctets(1.10)	Counter		介面收到資料的位元組數量。
ifInUcastPkts(1.11)	Counter		傳入 Unitcast 的封包數量。
ifInNUcastPkts(1.12)	Counter		傳入非 Unicast 的封包數目。
ifInDiscards(1.13)	Counter		進入封包被拋棄的數量。
ifInErrors(1.14)	Counter		進入封包因錯誤而被拋棄的數量。
ifInUnknownProtos(1.15)	Counter		進入封包因協定不明而被拋棄的數量。
ifOutOctets(1.16)	Counter		該介面傳出封包的數量。
ifOutUcastPkts(1.17)	Counter		傳出 Unitcast 的封包數量。
ifOutNUcastPkts(1.18)	Counter		傳出非 Unicast 的封包數量。
ifOutDiscards(1.19)	Counter		傳出封包被拋棄的數量。
ifOutErrors(1.20)	Counter		傳出封包因錯誤被拋棄的數量。
ifOutQlen(1.21)	Gauge		等待傳出的封包佇列長度。

ifSpecific(1.22)	ObjectID		物件描述的 MIB 參考定義。

16-8-3 位址轉譯群組

位址轉譯（Address Translation, at）群組包含單一個表格，提供網路位址和實體位址的對照，表格上有三個欄位：atIfIndex、atPhysAddress 與 atNetAddress，另外 ifEntry 為表格進入點，各管理物件的描述如表 16-10 所示。

表 16-10 at 群組之管理物件（1.3.6.1.2.3）

物件名稱	資料型態	讀/寫	描　　述
atTable(1)	Sequence of	NA	網路位址和實體位址的對照表。
ifEntry(1.1)	Sequence	NA	包含下面列出物件的對照。
atIfIndex(1.1.1)	Integer	是	對每一個特定對照的參考編號。
atPhysAddress(1.1.2)	PhysAddress	是	媒體相關的實體位址。
atNetAddress(1.1.3)	IpAddress	是	媒體相關的 IP 位址。

16-8-4 網際協定群組

網際協定（Internet Protocol, ip）群組包含測量系統進出的 IP 交通流量的網路計數器，ip 群組也包含若干格表格，提供網路階層路由和資料連結對照資訊，各管理物件的描述如表 16-11 所示。

表 16-11 ip 群組之管理物件（1.3.6.1.2.4）

物件名稱	資料型態	讀/寫	描　　述
ipforwarding(1)	Integer	是	1:表系統正轉送封包、0:表無轉送。
ifDefaultTTL(2)	Integer	是	預設 TTL 的值。
ipInReceives(3)	Counter		所有介面收到 IP 封包總數。
ipInHdrError(4)	Counter		因 IP 標頭錯誤而拋棄的輸入封包數量。
ipInAddrErroe(5)	Counter		因 IP 目的位址錯誤而拋棄的輸入封包數量。
ipForwDatagrams(6)	Counter		轉送封包的數量。
ipInUnknowProtos(7)	Counter		因協定不明而拋棄的封包數量。
ipInDiscards(8)	Counter		因緩衝器不足而拋棄的輸入封包數量。
ipInDelivers(9)	Counter		成功傳送給上一層的輸入封包數量。
ipOutRequests(10)	Counter		高層傳送到 IP 以供傳送的封包數量。
ipOutDiscards(11)	Counter		因緩衝器不足而拋棄的輸出封包數量。
ipOutNotRoutes(12)	Counter		因不知轉送路由而拋棄的輸出數量。

ipReasmTimeOut(13)	Counter		分段封包重組等待的最大時間。
ipReasmReqds(14)	Counter		需要重組而已收到的區塊數量。
ipReasmOKs(15)	Counter		成功重組的 IP 區塊數量。
ipReasmFails(16)	Counter		偵測出重組失敗的次數。
ipFragsOKs(17)	Counter		成功被切成區塊的封包數量。
ipFragsFails(18)	Counter		標示不可分割而需分段的封包數量。
ipFragsCreats(19)	Counter		成功分段的封包數量。
ipAddrTable(20)	Sequence of	NA	IP 位址表格。
ipRouteTable(21)	Sequence of	NA	IP 路由表格。
ipNetToMediaTable(22)	Sequence of	NA	IP 位址與實體位址之間的對照表。
ipRountingDiscards(23)	Counter		被捨棄的合法封包總數。

在 ip 群組中包含三個表格：ipAddrTable、ipTouteTable 與 ipNetToMediaTable。ipAddrTable 表格儲存與系統上使用的 IP 位址相關的資訊，它每一列以 ipAdEntlfIndex 作索引，包含和某一實體網路介面對應的單一個 IP 位址，ipAddrTable 表格內的管理物件如表 16-12 所示。

表 16-12 ipAddrTable 表格之管理物件（1.3.6.1.2.4.20）

物件名稱	資料型態	讀/寫	描 述
ipAddrEnty(1)	Sequence	NA	IP 位址表格的入口。
ipAdEntAddr(1.1)	IpAddress		這一列的 IP 位址。
ipAdEntlfInfex(1.2)	Integer		相對應介面號碼：ifIndex。
ipAdEntNetMask(1.3)	IpAddress		IP 位址的子網路遮罩。
ipAdEntBcastAddr(1.4)	Integer		IP 廣播位置最右邊的位元值，”1”。
ipAdEntReasmMaxSize(1.5)	Integer		可以重組之最大分段大小。

表 16-13 為 IP 路由表，也是由 ipRouteEntry 為表格進入口，包含 13 個管理物件，由這些管理物件的訊息可以觀察到路徑選擇的情形，尤其當該管理設備是路由器時，更顯得重要。

表 16-13 ipRouteTable 表格之管理物件（1.3.6.1.2.4.21）

物件名稱	資料型態	讀/寫	描 述
ipRouteEntry(1)	Sequence	NA	表格特定入口。
ipRouteDest(1.1)	IpAddress	是	路徑的目的位址。
ipRouteIfIndex(1.2)	Integer	是	下一路徑的本地介面識別碼。
ipRouteMetric1(1.3)	Integer	是	本路徑的第一個路由值。

ipRouteMetric2(1.4)	Integer	是	本路徑的第二個路由值。
ipRouteMetric3(1.5)	Integer	是	本路徑的第三個路由值。
ipRouteMetric4(1.6)	Integer	是	本路徑的第四個路由值。
ipRouteNextHop(1.7)	IpAddress	是	下一路徑的 IP 位址。
ipRouteType(1.8)	Integer	是	路由型態：其它=1、無效=2=、直接=3、間接=4。
ipRouteProto(1.9)	Integer		路由協定：其它=1、ICMP=4、RIP=8、OSPF=13、BGP=14。
ipRouteAge(1.10)	IpAddress	是	上次被更新或認證的時間計數。
ipRouteMask(1.11)	IpAddress	是	用來和目的位址作 AND 運算的遮罩位址。
ipRouteMetric5(1.12)	Integer	是	本路徑的第五個路由值。
ipRouteInfo(1.13)	ObjectID		描述此路由的 MIB 訊息。

表 16-14 為 IP 位址轉譯表格（ipNetToMediaTable），此表格和 at 群組的功能相同，當 at 群組物件被拒絕使用時，可用此表格代替。

表 16-14 ipNetToMediaTable 表格之管理物件（1.3.6.1.2.4.22）

物件名稱	資料型態	讀/寫	描　　述
ipNetToMediaEnty(1)	Sequence	NA	表格入口。
ipNetToMediaIfIndex(1.1)	Integer	是	相對介面的 IfIndex。
ipNetToMediaPhysAddress(1.2)	PhysAddress	是	實際位址。
ipNetToMediaNetAddress(1.3)	IpAddress	是	IP 位址。
ipNetToMediaEntry(1.4)	Counter	是	產生類型：其它=1、無效=2=、動態=3、靜態=4。

16-8-5 網際控制訊息協定群組

icmp 群組包含數個計數器來記錄**『網際控制訊息協定』（Internet Control Message Protocol, ICMP）**的運作情形，主要分為兩大類：紀錄輸入訊息（1 ~ 13）和紀錄輸出訊息（14 ~ 26），如表 16-15 所示。

表 16-15 icmp 群組的管理物件（1.3.6.1.2.5）

物件名稱	資料型態	讀/寫	描　　述
icmpInMsgs(1)	Counter		收到 ICMP 訊息的總數。
icmpInError(2)	Counter		收到 ICMP 有錯誤的訊息總數。
icmpInDestUnreachs(3)	Counter		收到 ICMP 無法到達的訊息總數。

icmpInTimeExcds(4)	Counter		收到 ICMP 溢時的訊息總數。
icmpInParmProbs(5)	Counter		收到 ICMP 參數問題的訊息總數。
icmpInSrcQuenchs(6)	Counter		收到 ICMP 來源抑制的訊息總數。
icmpInRedirects(7)	Counter		收到 ICMP 重導向的訊息總數。
icmpInEchos(8)	Counter		收到 ICMP 回應請求的訊息總數。
icmpInEchoReps(9)	Counter		收到 ICMP 回應回覆的訊息總數。
icmpInTimestamps(10)	Counter		收到 ICMP 時間戳記的訊息總數。
icmpInTimestampReps(11)	Counter		收到 ICMP 時間戳記回應的總數。
icmpAddrMasks(12)	Counter		收到 ICMP 位址遮罩要求的總數。
icmpAddrMaskReps(13)	Counter		收到 ICMP 位址遮罩回應的總數。
icmpOutMsgs(14)	Counter		送出 ICMP 訊息的總數。
icmpOutErrors(15)	Counter		因 ICMP 問題而無法送出的總數。
icmpOutDestUnreachs(16)	Counter		送出 ICMP 目的無法到達的總數。
icmpOutTimeExcds(17)	Counter		送出 ICMP 溢時訊息的總數。
icmpOutPramPribs(18)	Counter		送出 ICMP 參數問題訊息的總數。
icmpOutSrcQuenchs(19)	Counter		送出 ICMP 來源抑制訊息的總數。
icmpOutRedirects(20)	Counter		送出 ICMP 重導向訊息的總數。
icmpOutEchos(21)	Counter		送出 ICMP 回應請求訊息的總數。
icmpOutEchoReps(22)	Counter		送出 ICMP 回應回覆訊息的總數。
icmpOutTimestamps(23)	Counter		送出 ICMP 時間戳記訊息的總數。
icmpOutTimestampReps(24)	Counter		送出 ICMP 時間戳記回應的總數。
icmpOutAddrMasks(25)	Counter		送出 ICMP 位址遮罩訊息的總數。
icmpOutAddrMaskReps(26)	Counter		送出 ICMP 位址遮罩回應的總數。

16-8-6 傳輸控制協定群組

tcp 群組包含若干個管理物件來儲存有關 TCP 協定的統計和運作訊息，另外還有一個表格（tcpConnTable）儲存關於 TCP 連接的資訊，其管理物件的描述如表 16-16 和 16-17 所示。

表 16-16 tcp 群組的管理物件（1.3.6.1.2.6）

物件名稱	資料型態	讀/寫	描　　述
tcpRtoAlgorithm(1)	Integer		重新傳送演算法：1=其它、2=RTO、3=MIL-STD-177、4=Van Jacobsin。
tcpRtoMin(2)	Integer		重新傳送計時器的最小值。

tcpRtoMax(3)	Integer		重新傳送計時器的最大值。
tcpMaxConn(4)	Integer		可以支援的 TCP 連線總數。
tcpActiveOpens(5)	Counter		主動開啟連線總數。
tcpPassiveOpens(6)	Counter		被動開啟的連線總數。
tcpAttempFails(7)	Counter		發生連線失敗的總數。
tcpEstabRests(8)	Counter		發生過連線重設的總數。
tcpCurrEstab(9)	Gauge		從 Establish 或 Close_wait 到 Closed 狀態的連接數量。
tcpInSegs(10)	Counter		收到區塊的總數。
tcpOutSegs(11)	Counter		送出區塊的總數，但不包含重送。
tcpReturnsSegs(12)	Counter		重新傳送區塊的總數。
tcpConnTable(13)	Sequence of	NA	TCP 連接資訊表格。
tcpInErrors(14)	Counter		收到有錯誤的區塊數量。
tcpOutRsts(15)	Counter		送出有 RST 其號的區塊數量。

表 16-17 tcpConnTable 表格之管理物件（1.3.6.1.2.6.13）

物件名稱	資料型態	讀/寫	描　　述
tcpConnEntry(1)	Sequence	NA	表格入口
tcpConnStat(1.1)	Integer	是	連接狀態：1=Closed、2=Listen、3=Sys_sent、4=Sys_Rcvd、5=Established、6=Fin_wait、7=Fin_wait_2、8=Close_wait、9=Last_ack、10=Closing、11=Time_wait、12=delete TCB（R/W）。
tcpConnLocalAddress(1.2)	IpAddress		本地 IP 位址。
tcpConnLocalPort(1.3)	Integer		本端埠口。
tcpConnRemAddress(1.4)	IpAddress		遠端 IP 位址。
tcpConnRemPort(1.5)	Integer		遠端埠口。

16-8-7 使用者電報傳輸群組

udp 群組包含有關管理裝置的 UDP 連線及運作情形，其中有一 UDP 表格（udpTable）紀錄本機位址和連接資訊，其它管理物件都是一個記數器，紀錄特定的 UDP 交通流量資料。表 8-18 與表 8-19 為 UDP 管理物件和 UDP 表格內容。

表 16-18 udp 群組的管理物件（1.3.6.1.2.7）

物件名稱	資料型態	讀/寫	描　　述
udpInDatagrams(1)	Counter		傳入的 UDP 封包總數。
udpNoPorts(2)	Counter		沒有特定連接埠口的 UDP 封包總數。
udpInErrors(3)	Counter		傳入的錯誤 UDP 封包總數。
udpOutDatagrams(4)	Counter		傳出 UDP 封包總數。
udpTable(5)	Sequece of	NA	Udp 表格。

表 16-19 udpTable 表格之管理物件（1.3.6.1.2.7.5）

物件名稱	資料型態	讀/寫	描　　述
udpEntry(1)	Sequence	NA	Udp 表格入口。
udpLocalAddress(1.1)	IpAddress		本地 IP 位址。
udpLocalPort(1.2)	Integer		本端埠口。

16-8-8 專屬群組

另外，ITEF 也允許各家廠商申請登錄 MIB，廠商將自家的網路設備的管理資料登錄於 MIB 上，方便於不同廠商之間的網路設備管理，如圖 16-5 上的 private 管理群組（1.3.6.1.4）。一般購買網路設備時，廠商都會附帶 MIB 標準格式的物件管理資料，讓使用者可以複製到管理系統的 SNMP Manager 上。因此，不同廠牌之間的網路設備，才可以互相管理。也可說是，目前 Internet 上不論網路管理或其它資訊系統（如 DNS）也都採用 MIB 的管理模式。

16-9 SNMP 版本問題

SNMP 是屬於『**分散式管理協定**』**(Distributed Management Protocol)**。管理系統可以由 NMS（安裝有 SNMP Manager）單獨管理，或 SNMP Agent 各自獨立管理，也可以讓 NMS 和 SNMP Agent 共同管理網路。當系統是由 NMS 和 SNMP Agent 共同處理情況下，NMS 也可以透過請求（Inform-Request），和其它 NMS 之間互相取得被管理物件的資料。基本上，SNMP 通訊協定缺乏認證處理的能力，傳遞中的訊息容易被偽裝、竄改、偷竊等等。一般被管理設備都只能以密碼方式來確認使用者身分，對於傳送中的訊息就比較難達到保密的功能。因此，一般廠商都不提供 Set-Request 的命令，當然，也會減低 SNMP 監督控制的能力。

其實目前流行的 SNMPv2 並不相容於 SNMPv1，其中有兩個主要原因：訊息格式（Message Format）和協定操作（Protocol Operation）。SNMPv2 使用不同於 SNMPv1 的封包標頭（Header）

和協定資料單元（PDU），又 SNMPv2 增加了兩組命令（GetBulk 和 Inform）。雖然 RFC 1908 上定義兩種方法：代理仲介（Proxy Agent）和雙語網路管理系統（Bilingqual Network Management System），可以整合 SNMPv1 和 SNMPv2 協定，使其能互相支援管理，但一般系統還是捨棄 SNMPv1，而更新為 SNMPv2 較多，但是 SNMPv2 在安全性考量上爭議還是非常的多，許多廠商還是保持觀望的態度，而不敢大量投資發展。

最後，許多研究人員嘗試在不同的遠端管理和安全提案中找出解決方案，共同撰寫另一系列的 RFCs，就是所稱謂的 SNMPv3。這些 RFCs（2271 - 2275）也經 IETF 認定為標準，也就是說，一般大眾都可以取得該標準，來從事安裝並檢討該標準的優異點。SNMPv3 增進了下列功能：

(1) 奠定 DES、MD5、或其它認證協定上的安全模組。

(2) 定義及制定查閱瀏覽的存取控制模組。

(3) 重新定義某些 SNMP 的觀念和術語。

雖然 SNMPv3 在市場上享有某種程度的優越性，但該協定還是非常新穎，而想要該網路被各廠商採用，可能還需花費一段長的時間，其主要原因是成本效益比是否值得，至於那些較嚴格的網路設備，它們會升級到新的協定，然而並不是每個人都會決定要升級現有設備來支援 SNMPv3，因此，整合的動作緩慢是可以預期的。

當網路設備愈來愈多時，網路管理人一直試圖找出比較完整的管理工具，但又牽涉網路設備製造商繁多，各有各自管理方法，如採用標準協定製作，又涉及到安全性的問題，的確是十分棘手的問題。記得十幾年前，各家廠商致力將 SNMPv1 協定植入網路設備，當時作者教導網路管理人員有關 SNMP 通訊協技術，首先，我就請學員以 SNMP Manager 擷取實驗室裡路由器的路由表，並重置路由器，所有學員都能如願達成目的，再請學員將 IP 位址設定到自己單位的路由器，也做同樣的動作，所有學員的臉都綠了（嚇死了），大家異口同聲決定要將 SNMP 功能關掉。SNMP 一直是被認為安全性最大的漏洞，但是大家一直追求網路傳輸速率要快、可靠性要堅固、安全性要提昇的同時，網路管理員更應注意網路是否易於管理，才能將網路隨時保持最佳狀態，這必須管理人員隨時觀察、統計分析網路的各種係數（狀況），如此必須仰賴 SNMP 的功能，因此，研習 SNMP 通訊協定是所有 Internet 網路管理員不可或缺的基本技能。

16-10 ASN-1 資料型態

SNMP 主要的訴求是希望在異質電腦環境下，建構一個共通的管理協定，但不同電腦之間對於資料格式的表示可能不會完全相同，會導致資料的認定發生錯誤。因此，在 SNMP 協定上 Manager 和 Agent 之間所傳遞的訊息，不論是物件識別值（Object Identifier）或數值（Value）都使用 ASN-1 編碼，以解決不同電腦之間資料格式宣告不同的問題，本節將針對 ASN-1 的原理與編碼技術加以介紹。

當兩端使用者之間傳遞訊息，如果只使用到檔案資料傳送時，無論傳送中的訊息是文字檔案（text file）、二進位檔案(binary file)或其他多媒體檔案。接收端只將收到的訊息重新再表現出來，並未將訊息內的每一資料拿出來處理。這種運作方式，兩端使用者只要協議使用何種格式傳送即可，不需要表現層特殊的處理（請參考圖 16-2）。

但對於較高階的網路應用上，也許並非如此單純。譬如，提款機和銀行主機電腦連線，目前各個銀行之間都有跨行連線，客戶在任何一家銀行的提款機都可以領到錢。例如客戶在郵局提款機前提領華南銀行內的存款，由提款機送出（華南、帳戶 A、-10000），表示要由帳戶 A 上扣除 10000 元。其運作方式是由提款機呼叫銀行主機電腦內提款處理程式，提款處理程式會驗證帳戶 A 內存款是否超過 10000 元，並扣除該款項。這表示由提款機送一個處理程式的參數（parameter）給主機電腦。倘若提款機和主機的電腦不同品牌，作業系統也不相同，由於提款機所傳送的參數資料格式表示不相同，雙方可能產生嚴重的誤解。譬如，有可能提款機要求扣除 10000 元，因資料格式表示不同，主機電腦只扣除 100 元，但他回應確認訊號給提款機，提款機竟給客戶 10000 元。在兩端傳遞各種訊息當中，沒有發生任何錯誤，但由於雙方資料格式不同，而產生嚴重的錯誤，並且這種錯誤又不容易被發現。

在電腦資料處理中，我們將一些變數宣告成某一資料格式，例如整數、實數、字元、字串等等。任何一種資料格式的變數都有預留適當記憶體空間讓它存放資料，例如一般整數（Integer）預留 2 Byte 的空間，我們稱之為**『抽象資料格式』(Abstract Data Type)**。當我們在程序呼叫（Procedure Call）時，會傳遞某些參數（Parameter）給程序（Procedure）執行，或由被呼叫程序傳回參數時，都是以某一種資料型態的變數來作為參數的傳遞或回應。如果僅就本地程序呼叫，因為變數的宣告格式相同，大致上不會發生什麼問題。但如果跨越遠端電腦，執行**『遠端程序呼叫』(Remote Procedure Call)** 時，雙方電腦上對於抽象資料宣告的格式就不一定會相同。如圖 16-8 所示，被呼叫程序可能在不同類型的電腦（或作業系統）上執行程式，由於各類電腦對抽象資料格式的宣告可能不同，因此程序呼叫所攜帶的參數認定也不會相同，因此會產生上述資料格式的誤解。

為了克服抽象資料格式在各個電腦上宣告不同而產生的錯誤，我們必須定義一種標準的資料格式。讓電腦在呼叫遠端程式時，所傳送的參數（Parameter）都用該標準格式來宣告，才不會發生資料判別的誤解。早期 ISO 制定一個標準格式，稱之為**『抽象語意表示法』(Abstract Syntax Notation - One, ASN-1)**，期望各個電腦之間傳送資料都用 ASN-1 格式宣告。如圖 16-9 所示，遠端程序呼叫所傳的參數用 ASN-1 宣告，遠端電腦上的對應程式也使用 ASN-1 宣告，才不會發生資料格式的誤解。

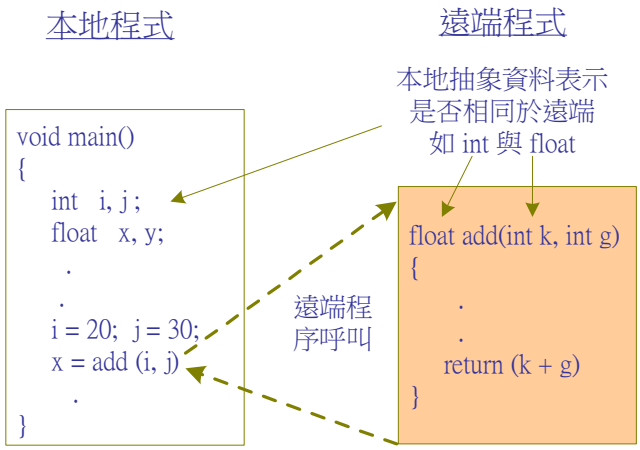


圖 16-8 遠端程序呼叫之參數資料格式

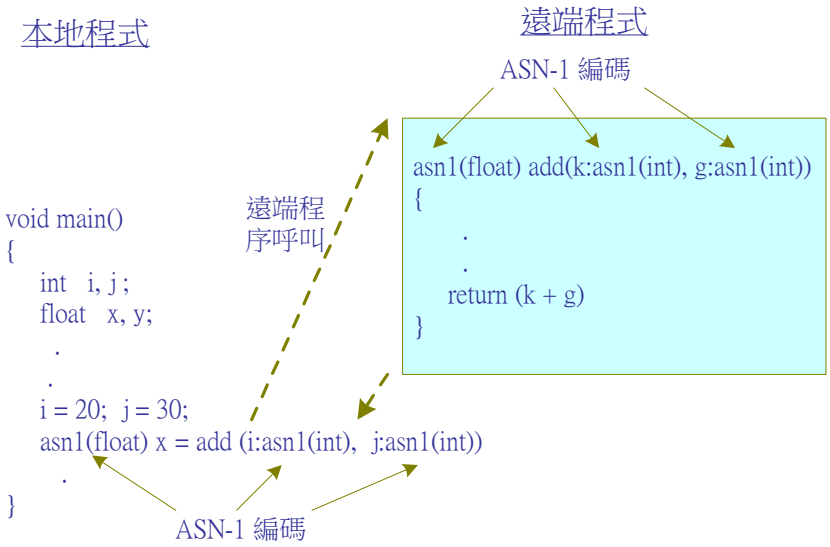


圖 16-9 遠端程序呼叫使用 ASN-1 資料格式

我們瞭解 ASN-1 是針對**『資料格式』(Data Type)**宣告的標準化，並非檔案格式(File Type)。ASN-1 是應用在電腦之間資料存取的標準化，傳送端的資料經過編碼器轉換成標準之 ASN-1 格式再發送；對方接收到後，再將其解碼還原成原始資料，如圖 16-10 所示。在 ISO 8824 中，定義 ASN-1 的語法，ISO 8825 則說明編碼規則。

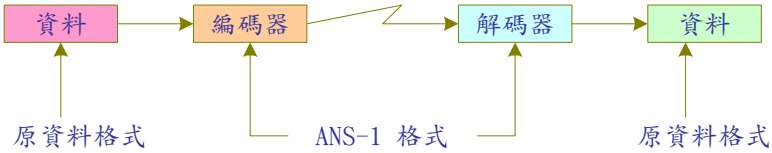


圖 16-10 ASN-1 資料編碼與解碼

如欲將傳送中的資料以 ASN-1 格式編碼，我們必須考慮下列兩點：(1) 定義一套語言來描述各種資料型態；(2) 另定義一套編碼規則，將資料型態轉換為數字碼，以作為傳送。ASN-1 將資料型態歸類為通用（universal）、應用（application）、指定文字（context-specific）、以及專用（private）等四類。對於每一種類型又可區分為基礎型態（primitive type）和結構型態（construct type）。其中通用（universal）型態是由 ASN-1 標準所定義，應用（application）型態是由其他標準所定義的，專用（private）型態是由各企業組織所指定，指定文字（context-specific）型態係自由指定。在這裡我們只針對通用型態介紹，因為其它型態目前甚少使用。

16-10-1 ASN-1 基本資料型態

如同一般程式語言一樣，ASN-1 的資料型態也可區分為基本型態（Primitive type）和複合資料型態（Construct type）兩大類。基本資料型態如表 16-20 所示，其中包含：整數、浮點數、字串等等，以下分別介紹各種基本資料型態的宣告方式：

表 16-20　ASN-1 基本資料型態

基本型態	意義
INTEGER	隨意長度的整數。
BOOLEAN	True (1) 或 False (1)。
BIT STRING	零或多個位元組成的串列。
OCTET STRING	零或多個無號位元組組成的串列。
NULL	位置保留元。
OBJECT IDENTIFIER	定義物件名字的型態。

(A) BOOLEAN 語法

用來描述真值表之中的真（True）或非真值（False）。

BooleanType ::= BOOLEAN

BooleanValue ::= TRUE | FALSE

(B) 整數（INTEGER）語法

用來描述整數值，其產生規則（production rules）如下：

IntegerType ::= INTEGER | INTEGER{NamedNumberList}

NamedNumberList ::= NamedNumber | NamedNumberList, NamedNumber

NamedNumber ::= identifier(SignedNumber) | identifier(DefinedValue)

SignedNumber ::= number | -number

IntegerValue ::= SignedNumber | identifier

例如：下列合乎 ASN-1 的語法定義：

Version ::= INTEGER{windows(1), unix(2)}

MyVersion ::= unix

我們在定義 Version 的整數資料型態時，包含 windows 和 unis，其相對值為 1 和 2。而將 MyVersion 的起始值設定為 unix，其相對值即是 2。

(C) 字串（OCTET STRING）語法

用來描述任意長度的字串，可描述二進位或十六進位字串。產生規則如下：

OctetStringType ::= OCTET STRING

OctetStringValue ::= binary-string | hexadecimal-string

(D) 空值（NULL）語法

用來描述保留位置。產生規則如下：

NullType ::= NULL

NullValue ::= NULL

(E) 位元字串（BIT STRING）

用來描述二進位元字串。產生規則如下：

BitStringType ::= BIT STRING

BitStringValue ::= {0 | 1}

(F) 物件識別碼（OBJECT IDENTIFIER）語法

一般我們描述某一個物件，是描述該物件位於『物件描述樹』(Object Identified Tree)的節點位置。物件描述樹是整個環境(或系統)裡所有物件的組織結構，物件識別碼是標示出該物件在物件描述樹的相關位置。由這個相關位置，我們也可以了解這個物件和其他物件的關聯關係。如圖 16-5 為簡單網路管理協定(Simple Network Management Protocol, SNMP)的協定標準架構。在 internet (1) 以下，又可分為 directory (1)、mgmt (2)、experimental (3)、private (4)、security (5) 及 snmpv2 (6)。

我們用 ASN-1 描述 tcp 物件可以有下列三種方式表示：

(1) tcp OBJECT IDENTIFIER ::= {.1.3.6.1.2.1.5}

(2) tcp OBJECT IDENTIFIER ::= {.iso.org.dod.internet.mgmt.mib-2.tcp}

(3) tcp OBJECT IDENTIFIER ::= {iso(1).org(3).dod(6).internet(1).mgmt(2). mib-2(1).tcp(5)}

16-10-2 ASN-1 複合資料型態

在 ANS-1 上所定義的複合資料型態(construct type)其實和 C 的結構(structure)、Pascal 的記錄(record)非常類似。基本資料型態可組合產生不同的複合資料型態，如表 16-21 所示，ASN-1 有五種型態。

表 16-21 ASN-1 複合資料型態

複合資料型態	意義
SEQUENCE	有序(order)串列型態，含不同基本資料型態。
SEQUENCE OF	有序串列型態，單一基本資料型態(如 array)。
SET	無序串列型態，含不同基本資料型態。
SET OF	無序串列型態，單一基本資料型態。
CHOICE	由型態串列中產生聯集的型態。

我們以序列(SEQUENCE)的語法宣告作為例子說明，它的宣告方式和程式語言中的 structure 很類似，譬如，假設只有四個欄位，可以定義如下：

```
Address ::= SEQUENCE
{
        street PrintableString,
        number Integer,
        city PrintableString,
        zip Integer
}
```

當然，在複合型態中的元素也是 ASN-1 的基本型態或複合型態。如上例之中，PrintableString 及 Integer 都必須按照基本型態宣告。其他 SEQUENCE OF 和 SET 的語法也都大同小異，我們不再另述。

16-11 ANS-1 編碼規則

ASN-1 的編碼規則也是採用一般『**基本編碼原則』(Basic Encoding Rule, BER)** 方式。其主要功能是定義 ASN-1 資料型態的值如何轉換為位元序列的排列方法，以作為傳輸使用，接收端也能依照此方法正確解碼。編碼規則是遞迴式，所以結構式物件 (複合資料型態) 的編碼為其擁有之基本物件 (基本資料型態) 編碼的連結。依照此種方法，所有物件編碼都可以歸類為基本物件編碼的串列。

BER 定義之中，不論基本物件或結構物件的值都是由四個欄位組成：(1) 識別碼，(2) 資料欄位的位元組 (Byte) 長度，(3) 資料欄位，(4) 結束欄位 (若資料長度未知)。結束欄位是針對內容並未定義完全的複合資料型態，如果資料長度都已知道，就不需要該欄位了 (如 SNMP 使用)。以下我們分別說明各個欄位的功能，如圖 16-11。

圖 16-11 ASN-1 編碼格式

16-11-1 識別碼欄位

如圖 16-12 所示，識別碼欄位中又區分為三個子欄位：(1) 標籤、(2) 資料型態、 (3) 數字。標籤顯示該編碼資料的類型 (通用、應用、指定文字或專用類型) 以兩個位元來區分。資料型態子

欄位如為 1 表示資料為基本型態；0 為複合資料型態。數字子欄位表示該資料是屬於資料類別中的哪一種資料型態。

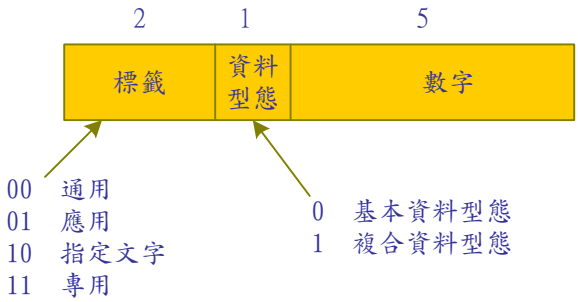


圖 16-12 ASN-1 BER 之識別碼

我們以通用類別（UNIVERSAL，00）來說明編碼的情況（因為在 SNMP 協定裡只用到通用類別），在通用類別之中有不同的資料型態（整數、字串等等）。表 16-22 為各種資料型態在數字子欄位的值。代碼中 11 ~ 15 及 28 ~ 31 保留。

表 16-22 通用類型的資料型態

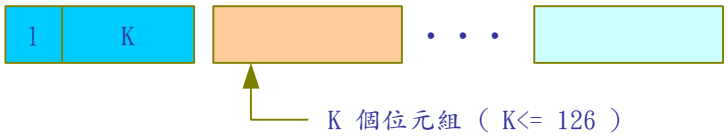
代碼（數字）	資料型態
1	BOOLEAN
2	INTEGER
3	BIT STRING
4	OCTET STRING
5	NULL
6	OBJECT IDENTIFIER
7	OBJECT DESCRIPTOR
8	EXTERNAL
9	REAL
10	ENUMERATED
16	SEQUENCE and SEQUENCE OF
17	SET and SET OF
18	NumericString
19	PrintableString
20	TeletexString
21	VideotexString
22	IA5String
23	GeneralizedTime
24	UTCTime
25	GraphicString
26	VisibleString
27	GeneralString

16-11-2 長度欄位

長度欄位是標示於緊跟後面資料內容的長度，有下列三種情況：

(1) 資料內容長度小於 128 Byte，只要一個位元組的長度欄位就足以表示。

(2) 資料長度大於或等於 128 Byte，長度欄位（1 Byte）不足於表示，必須以多位元組表示時。第一個位元組紀錄該長度總共需要幾個位元組（下圖中的 K），自第二位元組起，將長度之數值自高位元到低位元依序放置。



(3) 資料內容不定長度。資料內容不定長度時，我們就無法用長度欄位來描述內容的長度，而將長度欄位全部設定為 1，且在資料欄位的後面加入兩個結束記號，結束記號是用一個位元組，各位元全部設定為 0。

16-11-3 資料欄位

資料欄位編碼依照各個資料型態而定，以下說明幾種資料型態的編碼方式：

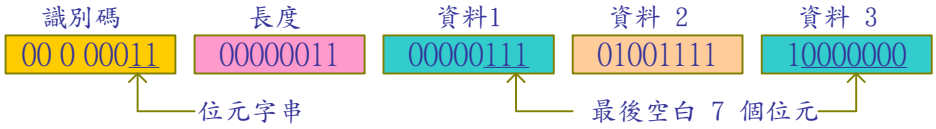
(A) 整數（INTEGER）

整數編碼以該整數之 2 的補數來編碼，小於 128 的正整數需要一個位元組長度，小於 32768 需 2 個位元組，以此類推。正負號位元組先傳送，自高位元至低位元，每八個位元為一組，依序置放。如：Integer 49 編碼結果如下：(標籤 = 00 表示通用類型)、(資料型態 = 0 表示基本型態)、(數字 = 2 表示整數)。(長度 = 1 表示資料 1 位元組)、(資料 = 49)

(B) 位元串列（BIT STRING）

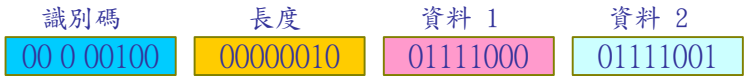
位元串列的資料編碼和原來資料相同，但最主要問題在於長度的表示方式。長度欄位表示後面資料位元組的長度，而非位元的長度。但當位元資料依序放置在資料欄位中，資料欄位後面會剩下

一些位元欄位空白，因此在長度後面必須緊接著一個位元組，主要用來說明最後資料位元組剩下幾個位元欄位未填放。如 Bit String '01001111' 9 個位元串列，它的資料編碼為 07、4F、80（16 進位），其結果如下：



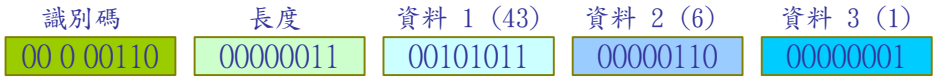
(C) 位元組字串（OCTET STRING）

位元組串列比較容易，串列字元以標準的方式由左至右傳送。如：Octet String "xy" 之編碼如下：(x, y 以 ASCII 碼表示)



(D) 物件識別碼（OBJECT IDENTIFIER）

物件識別碼編碼都是以整數串列表示。我們以簡單網路管理協定（SNMP）為例來說明其物件編碼情況，例如，internet 物件的識別碼是 { 1，3，6，1 }(如圖 6-11 物件識別樹)。在 SNMP 的物件描述識別之中，第一個數字永遠是 0、1、或 2；第二個數字要小於 40（根據 SNMP 規格，將無法辨識第 41 類)。因此，我們可以將前面兩個數字組合成一個數字編碼，只佔一個位元組的編碼資料，第一個數字稱為 a，而第二個數字稱為 b，其編碼值為(40*a + b)，依照組合編碼 internet 的第一個資料編碼為 43（40*1 + 3），其餘第三個數字以後，每一個數字填入一個資料欄位內。依照慣例，物件識別碼超過 127 的數字，會編碼成多個位元組，第一個位元組的高位元設定為 1，位元組總數存在另 7 個位元。Internet 物件編碼結果如下：



(E) 序列（SEQUENCE）

序列是屬於複合資料型態，它的編碼實際上就是將它所包含的基本型態，依序編碼組合而成。我們以地址結構來說明編碼的情況，其內部的子變數資料型態都假設已宣告：

SEQUENCE { Name "Tain-shou Nien",

Street "Hwang Po 1st ",

Number 1,

City “Feng-Shan”,

Zip 809

}

編碼方式如下：(編碼數字以十六進位表示)

1. SEQUENCE 的識別碼為 30，長度 30 是以下五筆變數資料的總長。(30，30)

2. Name 是 PrintableString 型態，識別碼 13，字串長度為 0E。(字元以 ASCII 碼表示)

3. Street 是 PrintableString 型態，識別碼 13，字串長度為 0C。

4. Number 是 INTEGER 型態，識別碼 02，內容長度為 1。

5. City 是 PrintableString 型態，識別碼 13，字串長度為 09。

6. Zip 是 INTEGER 型態，識別碼 02，內容長度為 2。

編碼結果：(編碼數字以十六進位表示)

(1) 30 30

(2) 13 0E 54 61 69 6E 2D 73 68 6F 75 20 4E 69 65 6E

(3) 13 0C 48 77 61 6E 67 20 50 6F 20 31 73 74

(4) 02 01 01

(5) 13 09 46 65 6E 67 2D 53 68 61 6E

(6) 02 02 03 29

習題

1. 試繪圖說明 SNMP 網路管理環境的架構圖。
2. 何謂『**要求/回應**』（**Request/Response**）？並請說明其特性。
3. 請繪圖說明 SNMP 協定 Get、Get-Next 與 Get-Response 命令的運作程序。
4. 請說明在 SNMP 協定上，執行 Get 和 Get-Next 兩個命令有何不同？
5. SNMP 通訊協定規範裡，在什麼情況下會發生 Trap 命令訊息？
6. 請參考 RFC-1448，說明 GetBulk 和 Inform 兩個命令的運作程序。
7. 為何 SNMP 必須使用兩個通訊埠口（UDP 161 與 162）？如果只使用一個通訊埠口會發生什麼問題？
8. 為何 SNMP 必須使用 UDP 協定傳輸？
9. 當 SNMP 所傳送的訊息超過 MTU 所限制的長度時，應如何處理？
10. 請利用您所擁有的 SNMP Manager 工具，擷取您電腦設備上的 MTU 數值。
11. 同上題，請擷取您電腦上的 ARP 位址轉譯表。
12. 同上題，請擷取您電腦上的 TCP 連接表，並作成表格說明其內容。
13. 同上題，請擷取您網路上路由器的路由表，並作成表格說明路由表的內容。
14. 試問如何利用 SNMP 擷取功能，來分析路由器的負荷情形？
15. 何謂「**SNMP 共同體**」（**SNMP Community**）？它在 SNMP 管理上扮演何種角色？
16. 請說明為何一般網路設備上的 SNMP 管理環境不提供 Set 命令的原因？
17. 何謂 MIB 資料庫，其如何描述管理物件？
18. 何謂『**抽象資料格式**』（**Abstract Data Type**）？
19. 請分辨 Pentium 系列和 Motorola 系列的電腦系統，對於資料在記憶體上排列有何不同的地方？並說明採用 ASN-1 編碼的原因。

20. 請說明當兩端資料宣告不同，執行遠端程序呼叫時，會發生什麼問題？

21. 何謂『**ASN-1**』？它如何解決遠端程序間呼叫，資料型態不一致的問題？

22. 請依照 ASN-1 編碼方法，將下列管理物件的識別碼編出其結果：

{iso(1).org(3).dod(6).internet(1).mgmt(2).mib-2(1).tcp(6).tcpRtoMin(2)}

{iso(1).org(3).dod(6).internet(1).mgmt(2).mib-2(1).udp(7).udpNoPorts(2)}

23. 請將下列資料以 ASN-1 編碼，並寫出其結果。(其中 Name、Section、Street 與 City 為 PrintableString 型態，Number 與 Zip 為 Integer 型態)

SEQUENCE { Name “Sun Yat-Sen University”,

Section “ Department of Electrical”,

Number 70,

Street “Lien-hai Rd”,

City “Kaohsiung”,

Zip 800,

}