乙太網路封包錯誤告警機制

符光恩
國家高速網路與計算中心
kwengenfu@nchc.org.tw

李進興
福雷電子
Vulcan_lee@aseglobal.com

摘要

由於乙太(Ethernet)網路的單價低廉，利用乙太網路當成企業內部網路的基礎架構已成為大多數的選擇。然而乙太介面數量龐大，乙太網路連結層的封包錯誤常被忽略，導致影響到網路的效能，嚴重時也會導致網路的癱瘓。本文將提出一種簡易的告警機制，用以長期檢查連結層的乙太網路封包錯誤問題，以閾值(Threshold)鎖定網路中封包錯誤過多的網路埠，並於第一時間掌握錯誤的來源，由於閾值檢查的週期不長，可以把握在網路癱瘓前的短暫時間發出告警，有助於立即的診斷。長期而言觀察這些錯誤的趨勢，可以在產生不利的影響前修正可能的錯誤。

關鍵詞：封包錯誤、閾值。

1.前言

隨著企業的成長，連接在企業內部區域網路中的設備也愈來愈多，而這些設備主要以乙太介面佔最多數，常見的設備包括伺服器、個人電腦、網路印表機及刷卡機等設備均會使用這種介面。不像專線的成本高，較容易受到監控，乙太網路的成本低廉，當事件發生時因礙於深入追查需要投入人力、物力，為加速排除問題，常以更換元件的方式處理，包括更換短線、網卡、設備。 實務上由於更換元件可以快速處理大部份的事件，也使事前預防的機制則較不受到重視，結果將導致潛伏的異常很容易被刻意的忽視，延後處理的時機。

其實疑似實體層和連結層的問題，可以由乙太網路錯誤封包看出一些端倪[1][2]，進而決定處理的方法。企業往往因受限於人力成本無法以人工普查的方式來找整個企業中的已知錯誤封包，導致缺乏預防機制。這類潛在的異常，也為企業埋下不可預期的危機，必需到異常變壞而成為事件後才會浮上枱面。

乙太封包錯誤告警的專案實施以後，以自動的方式將有封包錯誤的介面自動找出來，並加以分類，成為事前主動預防的機制，只要經比對閾值判定為異常，即開始受到監控，並安排處理時間，化解潛在的封包錯誤可能對企業所造成的損失。

整個告警機制是以業界最常採用的標準來實作，具有不限廠牌的特性。可設定每分鐘的閾值，異常時才會經由網路傳送訊息，具有節省網路頻寬的特性。因為是離線管理，並不限網路埠的數量具有高度的擴充性。該機制可以套用在乙太網路以外的介面，如 PPP、SDLC、FDDI、DS1、E1、PRI、Software Loop back 等，具有延展性極佳的特性。

1.1 研究的背景和動機

2003 年 12 月 22 日網路組接到員工抱怨：內部網路癱瘓。當時公司內部約有 60 台閘道器及 150 台交換器，約 3000 個乙太網路埠。經檢查發現該網段專屬閘道器的 CPU 運轉已達滿載，該網段對外完全無法連絡，但其他網段並沒有受到波及，判定為單一區域網路癱瘓事件。網路組約花了二個小時處理此單一區域網路癱瘓事件，處理的方式相當原始，包括重新啟動路由器，到現場逐一移除網路交換器上的短線，直到找到有問題的個人電腦。初步檢查到交換器上該網路埠有大量的 Ignore 封包，是導致這次區域網路癱瘓事件的原因。

好景不常，2003 年 12 月 31 日網路組再度接到員工抱怨：內部網路癱瘓。經檢查發現還是上次那台個人電腦造成該區域網路第二次癱瘓，這次花費 45 分鐘。2004 年 1 月 7 日第三度發生問題，仍是那台個人電腦造成該區域網路第三次癱瘓，這次花費 8 分鐘找到有問題的個人電腦。

整個過程中看似問題解決的速度愈來愈快，其實是因為網路組已經找到了處理這種事件的 workaround，並標示好該電腦使用的網路埠，隨時待命關閉。不幸的是，如果該電腦移到其他樓層，則仍需要花費 45 分鐘到二個小時，來找導致事件發生的禍首。

像以這樣的 workaround 來處理事件的方式存在每個企業之中，很明顯這種方式只能解決企業臨時的問題，長期來看反而是為企業埋下不可預知的危機，於是網路組針對解決乙太封包錯誤尋找可以事先告警的解決方案，以期主動發現潛在的問題。

1.2 研究的目的與方法

文中提出的封包錯誤告警機制，其目的在減少潛在的連結層的封包『異常』以避免『事件』的發生。至少要能達到每分鐘處理 3000 個左右的乙太介面，規劃此告警機制時必需具備以下的特色。

- 不限廠牌

- 節省網路頻寬
- 不限乙太網路埠數
- 具有擴充性
- 不需額外的費用

由於交換器是主要處理連結層訊號交換的設備，因此以交換器來檢查乙太網路的封包錯誤是很好的選擇。本系統主要利用 SNMP，以 MIB-II[5]、RMON[6]來建置，SNMP Trap[3]來告警。

2.相關技術和探討

2.1 Interface Group MIB

是 MIB-II 中的一個 Group[4] 主要是用於檢視每個介面的狀態，例如介面的輸入錯誤(ifInError: 1.3.6.1.2.1.2.2.1.14) 以及輸出錯誤 (ifOutError: 1.3.6.1.2.1.2.2.1.20)。通常輸入錯誤是指 runts、giants、no buffer、CRC、frame、overrun 以及 ignored counts 等各類輸入錯誤封包的總合[2]。而輸出錯誤則是所有輸出錯誤的總合。利用監控 ifInError、ifOutError 可以監控到所有的封包錯誤。

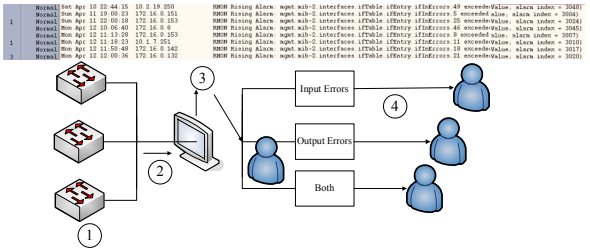
2.2 RMON MIB

RMON[6]設計的目標中，包括了離線管理、主動式管理、問題偵測 (Off line Management、Proactive Monitoring and Problem Detection) 利用這種技術可以配合閾值產生使用者自訂的 SNMP Trap，閾值的設定有二種型態: Delta、 Absolute。

例如 Delta Type 型的 Alarm 可以設定封包錯誤在 1 分鐘(取樣時間)內超過 100 個(Rising Threshold)錯誤封包即產生 SNMP Trap，以達到離線管理、主動式管理及問題偵測的目的，這有助於預報異常狀況並且不會佔用太多的網路頻寬。

3.設計架構

參考圖一的架構圖，是整個乙太封包錯誤告警機制的流程，其中最左方標示①者為 Device Level 主要以交換器為主，標示②者為設備發現封包錯誤超過閾值後所送出的 SNMP Trap，標示③者為 Network Management System Level 主要為網管主機，標示為④者為 Administrator Level，由管理者解決或由 Helpdesk 同仁協助解決需要現場才能處理的工作。



圖一 架構圖

3.1 規劃

如表一 規劃了三個階段和主要的工作。

表一：三階段工作及其工作內容

第一階段	準備工作 ● 交換器設定 ● RMON 資料收集 ● 進階資料收集 ● 分類
第二階段	處理 ● 以人工進行乙太網路的問題處理
第三階段	分析 ● 依錯誤原因、封包錯誤比例、封包錯誤型態分類及計算加權值 ● 閾值的決定

3.2 實作

3.2.1 準備工作

3.2.1.1 交換器設定

交換器必需支援 SNMP 的 RMON 及 MIB-II，利用這兩項技術可以由設備自行監控封包錯誤，以 Cisco 3548 為例，參考設定如下：

```
Rmon event 9010 log trap <community> description "High Input Errors" owner Ken
```

以上指令主要是設定編號 9010 的事件(Event)，當事件發生後會送出"High Input Errors"的 SNMP Traps。用同樣的方法可以設定可以送出"High Output Errors"的 SNMP Traps。

```
rmon alarm 3001 ifEntry.14.2 60 delta rising-threshold 100 9010 falling-threshold 90 owner Ken
```

以上指令主要是設定觸發的相關條件以及閾值。當監控的項目超過閾值(100)時會發觸發編號 9010 的事件, 設定條件如下：

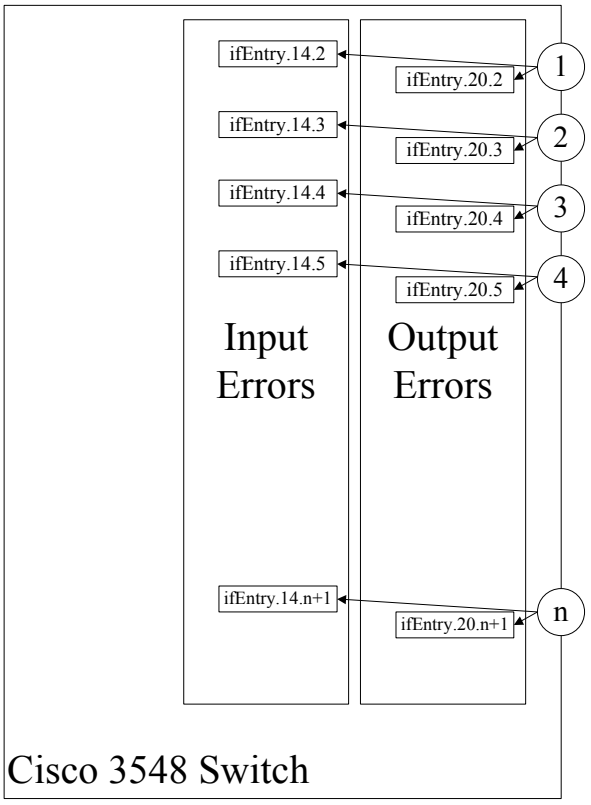
1. 檢查的項目是 ifEntry.14.2
2. 每 60 秒為週期進行檢查
3. 以 Delta 的方式比對閾值
4. Rising-threshold = 100，超過則觸發編號 9010 的事件以送出 SNMP Traps。

5. Falling-threshold=90

如圖二所示，由於每台交換器自行以 60 秒為週期檢查每個網路埠的閾值，以 Cisco 3848 為例，共有 Fast Ethernet 48 埠，以及 Giga Ethernet 2 埠，故 n=50。

以監控第二個 Fast Ethernet Port 為例，Input Errors 監控 ifEntry.14.3 (1.3.6.1.2.1.2.2.1.14.3)，Output Errors 為 ifEntry20.3 (1.3.6.1.2.1.2.2.1.20.3)。

由於監控採用的是 SNMP 中標準的 RMON-II MIB 故即使不是 Cisco 的設備也可以利用這種方式來監控。



圖二 架構圖

由於 RMON 離線管理的特性，檢查每個網路埠閾值的工作由交換器內部自行檢查，只有在超過閾值後才會以 SNMP Traps 回報，故僅佔用少量網路頻寬。這種由交換器自行檢查的分散式離線處理的架構，可以將運算的工作分散到每一台交換器的方法，故不會因為乙太網路埠數的增加而消耗網管主機的運算能力。

3.2.1.2 RMON 資料收集

在發現錯誤封包的來源後，可以針對有封包錯誤的網路埠進行更進一步的資料，收集的方式可以利用 RMON 統計的功能，以 Cisco 3548 為例，參考指令如下：

```
interface FastEthernet0/27
rmon collection stats 2027 owner Ken
rmon collection history 1027 owner Ken buckets 60
interval 1
rmon collection history 1127 owner Ken buckets 60
interval 60
rmon collection history 1227 owner Ken buckets 24
interval 3600
```

以上指令主要是 RMON 的方式，設定監控 24 小時的封包狀況，監控記錄會留在交換器端，可供必要時查詢。當確認乙太網路封包錯誤發生的時候才有必要設定，當異常解決後，這些監控的設定，便可以移除。

3.2.1.3 進階資料收集

為了能收集資料以供後續分析，在發現封包錯誤後，必需保存現有的資訊，以及收集使用者的資料以利核對及未來的追查。

交換器端的參考指令：

```
Show mac-address-table | in 0/27
Show interface FastEthernet0/27
Show Controllers Ethernet-controller FastEthernet
0/27
```

使用者端的參考指令：

```
date/t > test.txt
time/t >> test.txt
ipconfig /all   >> test.txt
route print >>test.txt
netstat -n   >> test.txt
ping 172.16.1.150   >>test.txt
ping 172.16.1.150 | FIND "TTL=" > NUL
IF ERRORLEVEL 1 tracert -d    172.16.1.150 >>
test.txt
netstat -e >> test.txt
arp -A >> test.txt
date/t >> test.txt
time/t >> test.txt
```

3.2.1.4 分類

當交換器發現設備出現乙太網路封包錯誤時，會以 SNMP Traps 的方式將事件送到網管軟體，可以經由適當的分類來處理這些異常，由於 Input Errors 和 Output Errors 的處理有很大的差別，所以有必要進行分類。

Traps 會附上相關的資訊，如下。

1. alarmindex
2. alarmVariable
3. alarmSampleType
4. alarmValue
5. alarmRisingTheshold

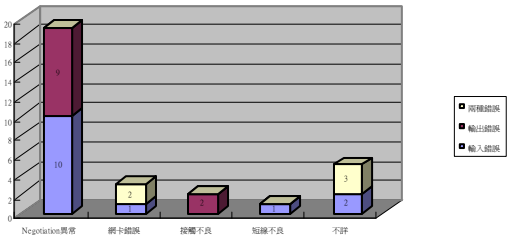
當網管主機收到事件後，將這個事件轉換成人員可以閱讀的告警，網管主機必需進行合併事件等處理，使網管人員容易閱讀，且數量不致過多。

3.2.2 處理

如圖三所示，Negotiation 異常佔 63.3%，這類的錯誤處理方式較為簡單工作人員不必到達現場利用簡單的電話通知或 E-MAIL 往返即可以處理。

處理過程中需要工作人員到現場排除的問題約佔 36.7%，在現場處理的過程中也發現大多數的問題可以經由封包錯誤的種類來判定，舉網卡異常為例可以發現這種網卡會產生大量的 Ignores 封包錯誤，或同時有大量的 Input/Output errors。

經驗的累積也是使判斷可以更加準確重要因素。本文的討論中以實際的經驗說明如何判定 80% 的封包錯誤問題的可能原因，包括經由速度(SPEED)、雙工狀況(DUPLEX)以及封包錯誤的種類來判定問題。



圖三 依錯誤原因分類

3.2.3 分析

3.2.3.1 針對發現的問題進行分析

分析 30 件有效樣本，Input Errors 的封包錯誤最多佔總數的 14 件，約全部的 46.7%， Output Error 11 件，約全部的 36.7%，而二種錯誤均有的佔 5 件，如圖三所示。

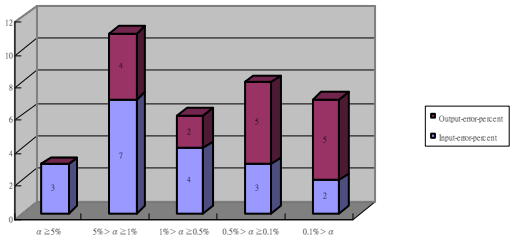
如由錯誤原因的角度加以分析，則 Negotiation 異常佔 63.3%、網卡錯誤 10%、接觸不良 6.7%。這三種主要的錯誤原因，佔全體錯誤的 80%，如果能針對這三種主要原因進行修正則可以解決大部份的問題，針對這三種主要錯誤原因的處理在本文後段的討論中有進一步的討論。

權重 α 的定義

α = error-percent = { input-error-percent | output-error-percent }

Input-error-percent=Δ(input errors)/ Δ(packets input)
output-error-percent=Δ(output errors)/ Δ(packets output)

錯誤封包可以依 α 值來決定嚴重性，如果錯誤封包佔所有輸入或是輸出封包的比重不高，其嚴重性就不高，α 值愈高表示佔的比重愈高，問題也就愈嚴重。在所有 35 件樣本中 input-error 共有 19 件樣本，output-error 共有 16 件，經分析可以發現，錯誤率的分佈如圖四所示。



圖四 依封包錯誤佔所有封包的比例分類

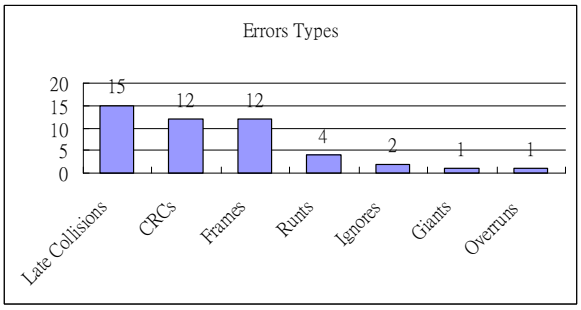
其中 $\alpha \geq 5\%$ 者共有三件，分別是 α=14.4%、α=9.3%，α=6.1%，算是比較嚴重的。

權重 β 的定義

β = {Ignores=n1,CRCs=n2, …, Late Collisions=1}
其中 n1, n2 為加權數。

在實務的經驗上不同種類的錯誤，會有不同的處理優先權，利用這個權重值可以提供判斷時決定事件重要性的依據。舉例而言，Ignores 封包錯誤曾對公司造成重大區域網路癱瘓的事件，因此其權重值設為 β=10。另外 Late Collisions 封包錯誤佔的數量多，但不會造成公司立即明顯的危害，故其權重值可以設為 β=1。

每個有效樣本會有一或多種錯誤型態，我們依封包錯誤的型態統計如圖五



圖五 封包錯誤的型態

其中 Late collision 為 Output Errors。其餘包括 CRCs、Frames、Runts、Giants、Ignores、Overruns 等均為 Input Errors。

在 Interface Group MIB 中有二個物件，分別是 ifInErrors 及 ifOutErrors，分別可以計算出網路埠的 input errors 以及 output errors 的總合。

3.2.3.2 閾值的決定

在 MIB-II 中 IfInErrors、IfOutErrors 分別代表了介面的輸入及輸出錯誤的總合。而經由計算單位時間所產生的錯誤量，可以推算出該介面的錯誤率並供該介面設定閾值的參考。

錯誤率的定義

Input-error-rate =delta(IfInErrors,t1, ifInErrors,t0) / (t1-t0) [1] output-error-rate =delta(IfOutErrors,t1, ifOutErrors,t0) / (t1-t0) [1]

單位時間產生少量的錯誤並不需要產生告警，一個分鐘產生一個 100 個錯誤比起一小時產生 100 個錯誤更有威脅性。在本文中是以最大錯誤率的百分之一來當閾值的基數。

在實務的經驗上我們了解到，乙太網路介面在一個 keep alive(約 10 秒)週期內如果有超過 5,000 個錯誤封包，則該網路埠會自動 shutdown[2]。

以此為標準決定最大錯誤率為：

max-error-rate = delta(5000,10, 0,0)/(10-0)=500 個/秒=30,000 個/分

閾值的基數為

Rising Threshold= max-error-rate/100 = 300 個/分。

在本改善中使用較嚴格的閾值標準約 100 個/分，約閾值基數的 1/3，主要是希望能收到較多的樣本。

4.討論

4.1 閾值、α、β與網管告警

在很多情況下錯誤封包超過閾值後，會送到網管主機如適當的分類那些是嚴重、主要、次要的告警，則很容易分出輕重緩急，處理到真正重要的告警。

文中提到的α和β值可以作為網管主機分類重要等級的加權數。例如同樣是每分鐘產生 100 個告封包錯誤的告警，但α=14.4%或β=10 是非常嚴重的告警，需要以簡訊方式通知工程師立即加以處理，但是 0.1%>α或是β=1 則可以當低優先權的狀況，不必急著處理，或者可以不必理會。

由於 RMON 所送出的 SNMP Traps 並沒帶，$\alpha$$\beta$值，如果有必要必需另寫程式自行計算。

4.2 Negotiation 等三種主要錯誤原因的處理

4.2.1 Negotiation 異常

Auto Negotiation 是目前乙太網路產品必備的能力，在實體線路接上以後，線路二端點的設備會自動溝通出適合雙方的速度和是否以全雙工對談。從實際的案例中發現，這類的問題最多，而交換器端可以觀測到只中有輸入封包錯誤或只有輸出封包錯誤。

由於大部份的問題是二個端點在 Negotiation 時仍有可能會有異常，故有時需要依賴手動設定速度及單/雙工的方式來克服這些異常，特別是伺服器如果沒有自動溝通到正確的設定，那很可能會因為此產生過多的錯誤封包，輕者反應時間變慢(單/雙工異常導致碰撞及重傳)，Throughput 降低(速度異常導致降速)，重者該網路埠自動 shutdown，主機將無法提供服務。

經由封包錯誤的情形、速度及單/雙工的狀態，可以簡單的確認是 Negotiation 的問題時，可以依據狀況進行不同的修正，例如伺服器超出閾值，則網路設備及伺服器端均固定設定；如果是個人電腦超出閾值，則網路設備及個人電腦端均設為自動偵測即可以解決，如仍無法解決問題，則判定非 Negotiation 異常，而以其他錯誤來處理。

4.2.2 網卡錯誤

乙太網路的技術已經非常精進，網卡的問題應不多，不知是否是筆者運氣太好，發現不少網卡有問題，除了有效樣本中有三台需要更換網卡或晶片外，公司還有一批筆記型電腦網卡所用的晶片有瑕疵並請廠商全數更換這批晶片。這個機制剛好可以用來找這批手提式電腦中尚未更換晶片的漏網之魚；不但如此連伺服器等級的網卡也被發現有產生 Ignores 封包的瑕疵，也在順利更換網卡後，將 Ignores 封包的問題解決了。

筆者在研究的背景和動機中曾提到一台個人電腦造成三次區域網路癱瘓的事件，就是因為網卡的問題而產生 Ignores 封包(α=14.4%)，如果當時這台個人電腦在別的區域網路接的是路由器(Router)就不會有問題了，但是天下就有那麼巧的事，那個區域網路的閘道器(Gateway)是一台橋接器(Bridge)用以串接一個在不同地理位置的網段，因橋接器經不起 Layer2 Ignores 封包的打擊， CPU 因為處理 Ignores 封包不及而滿載無法處理其他的工作。試想那台個人電腦是一台伺服器，並於每日清晨二點進行大量備份工作，那就不太好玩了。

4.2.3 接觸不良

在清查線路問題時有時感覺像是有是有靈異事件，因為封包錯誤告警機制非常準確的報告有大量的錯誤，依程序處理也以為線路有問題，可是將新線路取回，原線路接回去又發現沒有問題了，這種事情處理多了，居然發現一個共同點發生在同一個辦公室。

公司內湖辦公室曾經淹水，該區個人電腦各別使用獨立的網路埠，很多網路埠有大量的輸出包錯誤清一色為 Late Collision 的輸出錯誤封包因輸出錯誤多半是實體網路媒體(physical network media)的問題[1]，在了解是線路接觸不良的問題後，就可以節省人力及很多不必要的誤判，直接從清理接頭的方向下手處理。

4.3 擴充及延展性

由於使用的是業界 RMON、MIB-II 的標準，因此這種機制運作並不限廠牌，特別是也不限只解決乙太網路封包錯誤，這種方法也可以套用在其他介面，包括 PPP、SDLC、FDDI、DS1、E1、PRI、Software Loop back 等介面的封包錯誤告警，其延展性相當大；由於是利用離線管理，有問題才回報的機制，理論上並沒有設備數量的上限，因此其擴充性高。

5.結論

因過高的連結層封包錯誤可導致網路的效能降低，或在特定情形下導致網路的癱瘓，自動將封包錯誤的問題找出來確有其必要性。

而實作的結果也證實在 2004/3/1 開始執行後二個月的時間內，使得我們管理的觸角可以向下延伸至實體層及連結層，順利為企業解決多項問題並有以下成果

1. 解決了多台伺服器主機低效能、回應時間太長的問題
2. 順利發現一批筆記型設備網路的晶片組需要更換
3. 重要伺服器網卡需要更換
4. 網路線路因泡水導致接觸不良等問題
5. 有預防封包錯誤導和第一時間解決的方案

從短期而言可在第一時間掌握錯誤的來源，有助於立即的診斷；長期而言觀察這些錯誤的趨勢，可以在因錯誤封包而產生不利的影響前儘速修正可能的錯誤。

本系統目前已經在乙太網路上順利運作，因為使用的是 SNNP 的標準，未來可以移植到，如 PPP、SDLC、FDDI、DS1、E1、PRI、Software Loop back 等界面上實作。

這種方式可以在一分鐘之內利用 RMON 離線管理的特性查詢 3000 個以上的乙太網路埠的封包錯誤的狀況，足以證明可以應用在各類大型的網路上。

參考文獻

[1] Allan Leinwand (1992,November/December), Accomplishing Performance Management with SNMP. The Simple Times, Volume 1,Number 5

[2] Internetworking Troubleshooting Handbook, Second Edition. Cisco Systems, Inc.

[3] J. Case, M. Fedor, M. Schoffstall and J. Davin, "Simple Network Management Protocol (SNMP)", RFC 1157, May 1990

[4] McCloghrie, K. and Kastenholz, F., "The Interfaces Group MIB", RFC 2863, June 2000

[5] McCloghrie, K. and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets - MIB-II", STD 17. RFC 1213, March 1991.

[6] S. Waldbusser, R. Cole and C. Kalbfleisch, "Introduction to the Remote Monitoring (RMON) Family of MIB Modules", RFC 3577, August 2003.