IXIA-APTS:網路設備自動檢測機制之研發與建置

張明裕　　李昶清
南台科技大學資訊管理系

摘要

網路設備通訊協定程式設計人員在一般的網路協定的設計 (design)與建置(implementation)時,應該著重於程式語法的撰寫及除錯,但是其實大部分都在找執行程式中的錯誤問題 ,這時候常常就會發生用大部分的時間來做重複測試動作,通常有經驗的程式設計 人員可能較快可以找出問題,但是程式設計經驗比較不足的程式設計師就需要花上好一段時間才能測試出程式錯誤 ,及真正問題所在。也因此研究設計人員會浪費許多無謂的時間在重複測試實際建置完成的受測試機台韌體 ,而耽誤了原有程式完成的時程。

所以在程式設計的初期,就有測試程序(test procedure)這樣的一個設計,某一項通訊協定完整的測試程序可以讓程式設計人員很快地找出待測機台的效能或是問題所在 。測試程序是藉由對程式設計人員欲發展的程式做出一套正確的標準。而這一套標準完全是由程式發展者本身對程式的基本立意精 神列出的程序標準。經由這樣的一套測試序列程序,程式設計人員可以節省很多的時間。

在網路程式設計方面,由於程式的設計以及程式的修改完成之後 ,需要寫入程式碼進入硬體,在重新啟動硬體,然後藉由網路通訊協定交通產生器 (traffic generator)如 SMARTBIT,IXIA 來模擬網路上的各種狀況,在來對已寫入新的程式碼的路由器 ,交換器,來做壓力測試(pressure test),但是這樣的測試過程當中 ,卻花費了不少無謂的時間在操作受測機台的終端機控制 ,測試,結果比對,再調整受測機台的終端機控制 ,再測試的重複工作上面,其實有部分的工作室是可以被整合及自動化的 。

本論文的目的在於設計一個可以省下測試時間的一個自動測試系統 IXIA-APTS(Automatic Protocol Testing System),此系統是利用 IXIA 平台發展通訊協定測試的通訊協定的自動測試程序將寫好的測試程式透過 IXIA 環境執行對受測機台做一致性測試。在自動化測試方面,使用終端機介面軟體 Telix 的錄製鍵盤 I/O 功能來操控受測交換機臺的韌體調整 ,並且結合效能測試,一致性測試,達成自動化整合測試環境的建置。

關鍵詞: 一致性測試,網際網路,網路通訊協定檢測,網路設備效能測試

一、緒論

1-1、

隨著電腦資訊科技的進步,電腦的作業系統程式,網路系統程式…等等,變得更龐大更為複雜,相對的,整個程式的穩定度及可靠度就會漸漸的降低,而"測試"就會變得很重要,因為透過對受測機臺的測試可以知道此程式在實際上級執行上是否有錯誤,進而修正錯誤,然後讓程式順利運行。

實機效能測試是在包含各個不同的狀態下,根據以往經驗及實際需求所制定的規格書,以網路通訊協定交通產生器(traffic generator)產生個種不同的網路環境,來給予受測機臺(IUT)最嚴厲的測試。這樣的測試可以讓測試人員真正了解受測機臺的功能運作正常度以及耐力承受極限。這是在產品出廠之前最後的密集試驗以及出廠後客戶反映的功能問題解決,都必須要用到的測試方法。

本專案想要達到的目的,就是讓測試序列產生器的功能更形提高。以本論文為例:吾人欲使用IXIA400 內附的通訊協定測試產生語言,也就是IXIS-API,來產生符合測試 RFC 規格書中對受測裝置中通訊協定的壓力測試。

有許多的路由器,交換器此類的網路交通裝置,當中的通訊協定在研發過程需要經過修改,寫入硬體,然後使用通訊協定交通產生器(本論文使用IXIA1600),模擬受測環境產生協定封包測試受測機器。但是若經過觀察,不滿意其輸出結果,則要進行再次修改受測網路設備參數,再寫入軟體,再測試的重複工作。

本論文提出的解決方案是將效能測試流程自動化,同時整合一致性測試,減少大部分不必要的重複測試動作,進而節省測試人員的時間及人力成本,並且真正測試出受測機臺的實際效能水準。

這裡將使用測試的網路設備是交換機,而測試的初期主題為有防止廣播風暴功能的交換機在廣播風暴下的執行效能,以及開啟交換機防止廣播風暴功能下的執行效能,將兩者結果和 RFC 規格書做比較,並且觀察是否符合 RFC 的規格要求。

二、網路設備一致性自動化測試

2-1、網路設備測試規格

2-1-1、BMWG(Benchmarking Methodology)

在 IETF 裡有一個基礎方法工作小組(Benchmarking Methodology)簡稱為 BMWG,這個工作組的主要任務是製定一系列評估各種網際網路技術規格制定的建議;而在未來, BMWG 會更進一步對在這些技術上建置的系統與作業流程提出建議。

每一個提出的建議中都描述了所涉及的設備、系統或作業流程的方式;討論與這個類型相關的性能特性;並針對對這些特性加以描述的測試規格;再制定對這些測試進行過程的完整方法;最後提出了對測試結果及相關報告。

2-1-2、RFC2544 及 RFC2889

從 BMWG 製定的 RFC 及草案中,可以看出他們的一個制定測試方法準則。一般先針對被測對象製定一套專有名詞,然後用這套專有名詞對被測對象定義測試的方法。

BMWG 最先製定了 RFC1242,在這個 RFC 中對網路設備性能基準製定了一套術語,然後在與之配套的 RFC 2544 中製定了對網路互聯設備基準的測試方法。

RFC 2544 中討論並定義了一組可以用來反映網際網路性能特性的測試,此外還描述了報告測試結果的具體格式。

本論文參考 RFC 2544 的網路設備具體測試方法。針對所有被測的網路設備;稱為互聯網路設備(Network Interconnect Device),做測試的系

統性描述，從被測設備與測試設備的設置，一直到測試結果的記錄，都提出了完整的規格。

2-1-3 交換機測試項目

交換機作為企業網路的核心連接設備，它的性能是保障企業網路速度的主要標準。本論文使用網路交通產生器 IXIA1600 對受測交換機之防止廣播風暴功能做交換機性能中的 3 項主要指標進行測試。

進行性能測試的主要依據是 RFC2544 和 RFC2285，其中 RFC2285 是專門為測試交換機所制定的測試規格。

2-1-3-1．Fully meshed throughput:

作為衡量交換機性能最重要的指標之一，I/O 效能的高低決定了交換機在沒有 Frame loss 的情況下發送和接收封包的最大速率。在測試時，我們在滿負載狀態下進行。該測試配置為一對一(one-to-one)。

2-1-3-2．封包遺失率 Frame loss:

該測試決定交換機在持續負載狀態下應該轉發，但由于缺乏資源而無法轉發的訊框的百分比。訊框丟失率可以反映交換機在過載時的性能狀況，相對於廣播風暴等不正常狀態下交換機的工作情況 frame loss rate 具有明顯的差別。

2-1-3-3．封包轉送率 forwarding rates

該測試在測驗交換機在不丟訊框的情況下能夠持續轉發數據訊框的數量。

2-1-4、廣播風暴(broadcaststorm)

本論文選用廣播風暴測試作為測試系統初期達成測試目標,並擬以發送 ARP 封包作為廣播風暴測試。所謂廣播風暴(broad-caststorm)是指不希望發生大量網路廣播封包，同時壅塞在要經由網路傳送到所有的網路區段。一般而言，廣播風暴會因佔用大部分的網路頻寬，造成網路訊息的逾時。

Layer 2 的 Switch 並無法有效的阻絕廣播域(Broadcast Domain)如 ARP (Address Resolution Protocol)及 Win95/98 中大量使用的 NetBEUI 協定均大量使用廣播封包，因此就算 Layer 2 Switch 以 VLAN (Virtual LAN)的方式(虛擬網路)將經常要通訊的群組構成一廣播域(Broadcast Domain)來試圖降低 broadcast 封包對網路層的影響，但仍無法完全避免廣播風暴問題(同一個 VLAN 間仍會產生廣播風暴)

為防止廣播風暴發生，影響整體網路效能，故發展一種協定讓 Layer 2 的網路設備(Bridge、Switch)不會因實體線路接成迴圈而造成影響，共原理是應用最小擴張樹（Spanning Tree）的演算法，將每一個 Layer 2 設備定 cost，先根據 cost 選擇一個 Switch 為 Root，依演算法使整個網路設備間不會造成迴路，並自動 block 造成迴路的接線，如此一來，就能避免廣播風暴的產生。

2-2、通訊協定一致性測試

所謂通訊協定一致性測試是要測試出一個實作出來的產品(implementation)是否與他原先依據設計的規格(specification)相同,由於不確定建置完成的軟體是否和通訊協定規格符合,故必須使用一些方法驗證,而這個驗證的過程,就稱為通訊協定一致性測試。

本論文要測試的通訊協定已經被寫成韌體並且燒錄在交換機內部晶片。而一致性測試的內容及標準是以 RFC2544 及 RFC2885 為測試準則。

2-3、自動化測試對測試效率的影響

本論文重要的精神之一就是自動化,如序論所述,測試時間的節省,對於一個專業的網路設備測試中心來說,是十分重要的,因為通常受測的網路設備

動軋上百台,如果自動化測試流程沒有建立,相對的測試過程的時間及人力成本就要增加許多,反之,如果測試流程以然完整建立,節省的成本不言可喻。

三、IXIA-APTS 系統架構及規格

3-1、系統開發環境

本測試系統開發環境以 IXIA 網路交通產生器,Telix SALT-script,及 Layer2Switch 為主,用 IXIA 的測試程式開發環境來開發效能及一致性測 試工作。

3-1-1、IXIA 網路交通產生器

本論文運用網路交通產生器 IXIA- traffic generator 為測試程式開發環境,它提供高效能,多埠同時產生大量網路封包,並且可就結果輸出做分析。以下為其功能概述:

(1) 16 Card Modules:

共有 16 個插槽,每個插槽最多可以容納四個埠,每一個埠為 10/100mbps 傳輸量本論文程式使用到 2 埠

(2) 封包傳輸種類:

從 layer2~layer3 的各種封包傳輸種類皆備,而封包傳輸形式分為 unicast, broadcast,multicast 為主,本論文使用 broadcast 封包傳輸形式

(3)作業系統平台:

作業系統支援 windows95,windows98 及 windows2000,本論文程式使用 windows98 為作業系統平台

(4)IXIA Explorer 應用軟體:

IXIA Explorer 應用軟體為架設在 IXIA1600,作為圖形使用者介面(GUI),方便測試人員用此介面做封包傳輸設定以及接收測試結果(傳回封包數量),本論文使用此介面來接收測試結果。

(5)IXIA Tcl/Tk script

Tcl(Tool command langudge),工具命令語言,因為易學易用,所以被廣泛應用於,而在 IXIA server 上架有 Tcl-HAL,是為 Tcl script 在 IXIA traffic generator 上的編譯器,可以將完成的 Tcl script 經 IxTcl-HAL 編譯無誤之後藉由 IXIA traffic generator 執行

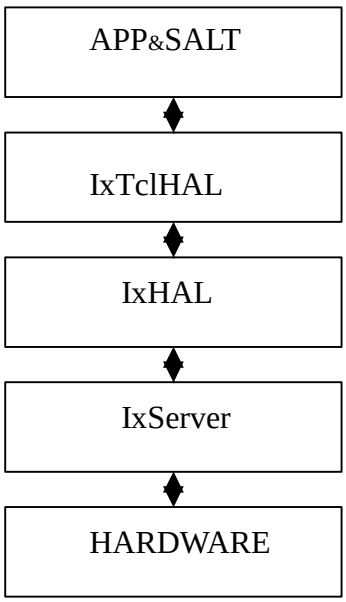
3-1-2 Telix

TELIX 是一種終端機介面的通訊軟體,使用在 DOS 和 Windows 作業系統,它提供了整合的通訊功能 一般此軟體會用在 BBS(Bulleting Board System) 撥接通訊上,或是 RS-232 的硬體控制上,本論文使用此軟體經由 RS-232 來作為交換機的 console 介面,控制受測機臺韌體的運作。

使用 TELIX 主要是因為其中更包含了特殊的 script 語言,方便使用者依照自己的需求,設計自動執行的巨集程式。而可以撰寫與通訊有關軟體,並且內建大量的函數方便程式寫作的 SALT(Script Application Language for Telix)-script 就是本論文會使用到的語言工具之一。

3-2、IXIA 系統架構

可參考圖一,其架構由下所述:



圖一 IXIA-APTS 架構

(1)C++APP:

將建置完畢的測試序列檢測軟體架設在 IXIA1600 網路交通產生器的最上層,以收取整合之效用。

(2)Telix:

使用 Telix 終端機介面的通訊軟體中 SALT-script 所附的鍵盤 I/O 動作的錄製功能。將交換機功能操作流程鍵入鍵盤並且錄製下來成為。slt 的 script 檔案,經過編譯之後,成為可執行副檔名為。slc 的檔案,並且放在 Tcl/C++主程式的子目錄底下,以供程式呼叫使用。

(3)IxTclHAL:

是用來接收上層應用程式的 Tcl/C++程式語言指令,並且轉換指令為硬體了解的機器命令語言的應用程式介面。

(4)IxHAL

是協助將收取到的指令設定資訊暫存在 Buffer 內,直到接收到上層要轉換這些資訊到 IxServer,才會做自動傳輸的工作。

(5)IxServer:

這是藉於控制網路交通產生器的電腦和網路交通產生器本身的伺服器,用來控制兩者之間的指令運作及資料流通。

(6)HARDWARE

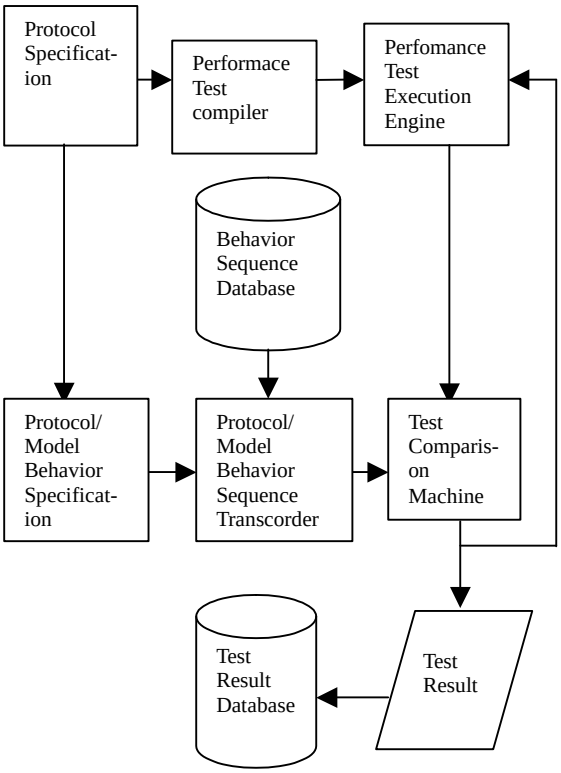
也就是指 IXIA1600 硬體主體,協助最終的測試產生的機器。。

四、IXIA-APTS 系統設計與實作

4-1、系統設計流程

第四部分為 IXIA-APTS 系統設計與實作,吾人先描述此測試系統所需之軟體,及硬體設備,併輔以圖形架構之規劃來了解完整自動測試的流程概要,自動測試流程主題初期以測試交換機的防止廣播風暴功能有無正常運作。

測試程式以防止網路廣播風暴功能關閉及開啟兩種模式測試各個不同等級的訊框處理能力。接下來便以虛擬碼的方式實現測試程式的演算法,以茲了解:



圖二 IXIA-APTS 協定檢測系統之功能模組與工作流程示意圖

詳細流程由上圖二所示:

(1) 受測通訊協定規格:

包括邏輯測試(一致性測試)以及壓力測試(pressure test)的受測的通訊協定可以先使用 EFSM 的格式先行大略描述出來,藉此方式可以讓使用者自己了解將要進行的測試的通訊協定的詳細規格。

(2) 通訊協定測試程序撰寫:

可以用方才描述的規格撰寫承 Tcl/C++的程式並且要注意一定要使用 IXIA 提供的 API 函式庫,以及結果比較的 STL(Standard- Template Libray)中的"MAP<>"函數也要包含在標頭內以供編譯。

(3) 編譯自動化測試程序:

這裡使用 IXIA 所提供的程式編譯器(compiler)來執行編譯,編譯過程中須把將使用到的

SALT-script 先行使用 Telix 的編譯器編譯完成之後，以供主程式使用。

(4) 測試執行引擎:

在測試執行階段,首先必須把產生出來的測試序列轉換成為可在 IXIA1600 執行的 Tcl/C++程式碼,這個部分是剖析器(parser)的工作,再來就轉換到 IXIA1600 的硬體部分,進行測試序列正式執行的工作。

(5) 通訊協定模組行為描述:

另一方面,必須把實際受測機台的所有行為詳細的條列出來,並且用 EFSM 的語法(c or psudo code)來表示,這樣的做法才以利下面的通訊協定序列資料轉換的步驟,並且儲存方式教不複雜。

(6) 通訊協定序列轉換:

這是為了要和 IXIA1600 產生的測試序列結果做比較,而進行的轉換。

(7) 測試比較機制:

最重要的一致性測試的精髓所在,也就是測試序列比對部分,必須要由這樣的一個測試比較機制來執行,當比較結果不符合規格要求時,就使用事先寫好的 Telix 的 SALT-script 的執行檔來代替人工操作交換機的 console 介面待自動操作調整完畢後再實施一次交換機防止廣播風暴的效能測試及一致性測試,重複此步驟直到達程規格要求的標準之後才跳出程式。

這裡使用 STL(Standard- Template Libray)中的"MAP<>"函數資料結構來實施比較功能。

(8)測試結果產出:

最後執行測試序列比較的結果將展現於表格上面,並且儲存到資料庫裡以供參考。

4-2、演算法及程式

這裡使用 TCL(Tool Command Language)和編譯完成的 SALT-script 為語言工具,以下為程式虛擬碼:

```
(01)# This package is required to have access to
(02)# the Ixia Tcl commands
(03)package req IxTclHal
(04)
(05)global one2oneArray
(06)
(07)#port configuration
(08)set txPortList    {{1 1 1}}     ;
(09)
(10)# High level API commands can use eather
(11)set rxPortList    {1,1,2}         ;
(12)
(13)# port list format or global array
(14)set portList       {{1 1 1} {1 1 2}}
(15)
(16)# Take ownership of the ports
(17)ixTakeOwnership              $portList
(18)
(19)# Set ports to factory defaults
(20)ixPuts "Setting ports to factory defaults..."
(21)if  [port  setFactoryDefaults  $chasId  $card
(22)$txport] {
(23)       errorMsg "Error setting factory defaults
(24)         on port $txport."
(25)            set retCode 1
(26)}
(27)
(28)# Writes port properties in hardware
(29)if {[ixWritePortsToHardware  one2oneArray]}
(30){
(31)    ixPuts "Error writing port to hardware"
(32)       set retVal 1
(33)}
(34)
(35)# Checks the link state of the ports
(36)if {[ixCheckLinkState    one2oneArray]} {
(37)   return -code error
(38)}
(39)
(40)logMsg "Configuring $chasId $card $txport
(41)--> $chasId $card $rxport"
(42)
(43)#Set default values percentage of total
(44)#packets sent to allow to be #lost before
(45)#declaring    packet loss
```

```
(46) ARP config -code 0
(47)
(48)# One frame size/frame rate test is called a
(49)#trial. The user may choose to run one or
(50)#more trials; the average result of each trial
(51)# will bepresented in the result file.
(52)ARP config -id 1 ;
(53)
(54)# total number of trials per frame size/frame
(55)#rate The approximate length of time frames
(56)# are #transmitted for each trial is set as a
(57)#'duration. The duration is in seconds; for
(58)#example, if the duration is set to one
(59)#second on a 100mbs switch, ~148810
(60)# frames will be transmitted. This number
(61)# must be an integer; minimum value is 1
(62)# second.
(63)ARP config -sequence 20 ;
(64)
(65)# duration of transmit during test, in seconds
(66)ARP config -type echoreply(0)
(67)
(68)# Start capture on Rx port
(69)ixPuts "Start capture..."
(70)if [ixStartCapture rxPortList] {
(71)    return -code error
(72)}
(73)
(74)# Start transmission on Tx port
(75)ixPuts "Start transmit..."
(76)if [ixStartTransmit txPortList] {
(77)    return -code error
(78)}
(79)
(80)# Checks whether transmission is done on a
(81)#group of ports
(82)if {[ixCheckTransmitDone txPortList] == 0} {
(83)  return -code error
(84)}
(85)ixPuts "Transmittion is complete."
(86)
(87)# Stop capture on ports
(88)ixPuts "Stop capture..."
(89)if [ixStopCapture rxPortList] {
(90)  return -code error
(91)}
(92)
(93)#use map<> to do conformace test(result
(94)#comparison); if result of comparison satisfy
(95)# the upboudary #then output else use the
(96)# SALT-script(config1.slc) to open the
(97)#switch's function prevent from broadcast-
(98)# storm and then execute the the C++
(99)# performance program again
(100)map<Key,T.Compare.Allocator>
(101){if map<> result not satisfy 1.execute SALT
(102) (2.execute the C++ performance program
(103)      if result not satisfy 1.execute SALT
(104)                              2.execute the C++
(105)                              performance test
(106)                              program
(107)                          .
(108)                          .
(109)                          .
(110)                          else-->output)
(111)          else--> output}.
(112)
(113)# Clear the ownership of the ports
(114)ixClearOwnership $portList
(115)
(116)# Log off user
(117)ixLogout
```

五、結論

本論文提出了一個以網路交通產生器 IXIA1600 **為主的自動化網路設備測試環境** IXIA-APTS,**不但結合了原有的效能測試的功能**,**更加上了符合使用者導向的一致性測試功能**,**讓測試人員能夠在短時間內以高效率的方式取得該受測網路設備的效能數據及資訊**.**而且是自動化操作**,**可以節省企業對於測試的時間及成本**.

在未來本系統雛形未盡完善之前,**還有發展的空間**,**主要有兩個方面**.**第一**,**在以** Telix **終端機軟體來控制網路設備的自動化過程**,**本論文是把各個網**

路設備的操作執行狀況事先錄製完成的方式,在主程式執行過程中再使用,但是亦可以程式產生器的方式自動產生.第二,儲存在資料庫裡的數據及資料可以發展成知識管理系統,使資料可以更有效的被運用以增加此系統的附加價值。

參考文獻

[1]Rui-Chi Wang，Protocol Conformance Testing Technology，**無線通訊技術專輯，第 79 期，**pp.17-24

[2]BMWG，RFC2544，IETF， pp3-4

[3]BMWG，RFC2289，IETF，pp28-29

[4] [ABC82] W. Richards Adrion， Martha A. Branstad，and John C. Cherniavsky，"Verification，Validation，and testing of Computer Software"，ACM Computing Surveys, 14(2):159-192，June 1982.

[5] [Dij68] E. W. Dijkstra，"Go to Statement Considered Harmful" (letter to the Editor), Communications of the ACM，11(3):147-148，March 1968.

[6] [GJM91] Carlo Ghezzi，Mehdi Jazayeri，and Dino Mandrioli，"Fundamantals of Software Engineering"，Prentice Hall，Englewood Cliffs，NJ，1991.

[7] [LR99] Chang Liu and Debra J. Richardson，" TestTalk，A Test Description Language: Write Once，Test by Anyone，Anytime，Anywhere，with Anything"，Technical report 99-08，Information & Computer Science，University of California，Irvine，February 1999.

[8] [Ric94] Debra J. Richardson，"Taos: Testing with analysis and oracle support"，In Proceedings of the 1994 International Symposium on Software Testing and Analysis