

行政院國家科學委員會專題研究計畫 期中進度報告

嵌入式網路通訊裝置評比技術與工具之研發--子計畫三:嵌入式網路通訊裝置核心與網路協定行為效能分析評比技術與工具之研發(中心分項)(1/2)
期中進度報告(精簡版)

計畫類別：整合型

計畫編號：NSC 98-2220-E-009-047-

執行期間：98年08月01日至99年07月31日

執行單位：國立交通大學資訊工程學系(所)

計畫主持人：曾建超

處理方式：本計畫可公開查詢

中華民國 99 年 10 月 03 日

行政院國家科學委員會補助專題研究計畫 成果報告

嵌入式網路通訊裝置核心與網路協定行為效能評比

技術與工具之研發

計畫類別：整合型計畫

計畫編號：NSC 98－2220－E－009－047

執行期間：98 年 08 月 01 日 至 99 年 07 月 31 日

執行單位：國立交通大學資訊工程學系(所)

計畫主持人：曾建超

計畫參與人員：李宗鴻、游家明、蘇育炆

成果報告類型：精簡報告

處理方式：除產學合作研究計畫、提升產業技術及人才培育研究計畫、列管計畫及下列情形者外，得立即公開查詢

☒涉及專利或其他智慧財產權，☐一年☒五年後可公開查詢

中 華 民 國 99 年 07 月 21 日

中文摘要

本兩年期計畫目標是要發展嵌入式網路通訊裝置核心與網路協定行為之效能分析評比的技術與工具。網路通訊裝置的通訊行為由核心行為與網路協定行為組成，因此本年度我們先進行「核心行為量測技術工具化」以及通訊行為整合分析之規劃。我們已完成記錄核心行為的封包導向之核心函式序列追蹤與量測技術，並將此追蹤與量測技術工具化，以及開放此工具的原始碼。此外，我們也搭配現有的網路通訊封包擷取工具，完成核心行為記錄與網路協定記錄的整併，以及通訊行為(含核心函式序列、核心事件與網路封包)合成的設計與實作，最後，我們也開發整合測試與評比的實施例，並進行本工具的實例測試。延續本年度成果，下一年度我們預計進行整體通訊行為的「整合分析工具」之開發與「核心行為量測技術」之優化，完成一套整合分析工具，以協助嵌入式裝置開發者診斷通訊行為的缺失或瓶頸，縮短研發週期。

關鍵字：嵌入式系統、網路協定堆疊、核心函式、核心事件、封包擷取工具

報告內容

一、前言

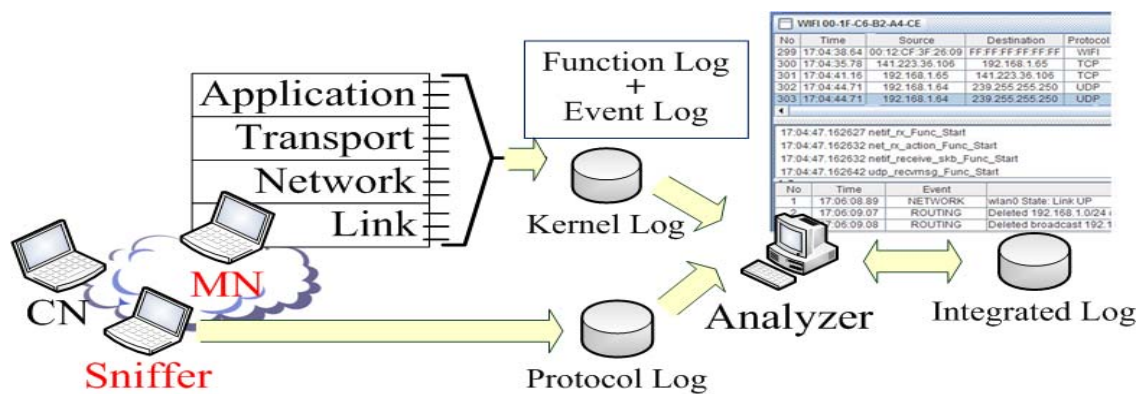
隨著電腦科技的進步與網際網路的興起，資訊產品的研發技術不斷地進步與創新，使得多種規格的無線/有線網路通訊裝置應運而生。此外，隨著網路通訊裝置的應用增加以及減少網路通訊裝置的體積、

重量與成本，嵌入式 (Embedded)網路通訊裝置日益普及，因此，有必要提供一套嵌入式裝置通訊行為的分析工具，以協助開發者診斷嵌入式裝置通訊行為的缺失和效能不佳(例如：傳輸延遲、反應時間以及封包遺漏等問題)的原因，縮短研發週期。

二、研究目的

本計畫之目的即是要發展嵌入式網路通訊裝置核心與網路協定行為之效能分析評比的技術與工具。一個嵌入式網路通訊裝置的通訊行為，包含基本的網路存取能力，以及其中的傳輸延遲、反應時間以及封包遺漏等效能指標。傳輸延遲、反應時間與封包遺漏會受到使用的網路類型、核心系統及網路協定等種種因素的影響而有所變化[1]。尤其當嵌入式網路通訊裝置在網路上漫遊(Roaming)期間，與各個無線網路基地台間進行交遞(Handover)時，從應用程式到網路協定，乃至作業系統內網路相關的核心元件，都會因應交遞而有所動作，所以也會造成傳輸延遲與封包遺漏。因此，有必要開發一套合適的輔助工具，協助嵌入式網路通訊裝置的開發者正確界定通訊行為的效能瓶頸，進而分析問題及改善缺失。

然而，目前尚未有一套完整的工具，能夠協助開發者深入瞭解嵌入式網路通訊裝置之通訊行為。雖然有類似的網路封包擷取工具(Network Sniffer)，卻僅能協助分析外顯的封包類型以及接收時間等，對於分析核心行為所引發的傳輸延遲等仍較缺乏。因此我們希望可以發展一套整合核心行為與網路協定行為的網路通訊裝置之效能分析評比工具。



圖一、系統架構

本年度計畫以「核心行為量測技術工具化與通訊行為整合分析之規劃」為目標，主要工作可以細分為(1)封包導向的核心函式序列追蹤與量測技術之研發(2)追蹤與量測技術之工具化與開放原始碼化(3)核心行為記錄與網路協定記錄整併之設計與實作(4)通訊行為(核心函式序列、核心事件與網路封包)合成之設計與實作(5)整合測試與評比規劃之研究，等五個工作項目。

三、系統實作架構

本計畫所設計的工具是讓使用者用來分析核心系統的核心行為以及網路協定行為，並獲得核心行為與網路協定行為彼此間互動的相關重要資訊，諸如核心函式呼叫的時間戳記(Time Stamp)、封包表頭(Header)欄位等，之後再利用這些資訊更進一步地分析核心系統在網路協定堆疊的處理過程中的效能，希望能夠藉此彌補目前網路封包擷取工具的不足。

圖一為本計畫所發展的整合分析工具之系統示意圖，圖中 MN 是欲觀察其核心行為的網路裝置，Sniffer 用來側錄 MN 傳送與接收的網路封包(網路封包可代表網路協定行為)，而 Analyzer 則是整合核心行為(Kernel Log)及網路協定行為(Protocol Log)兩種記錄，產生整併的紀錄(Integrated Log)，並進行網路行為分析。其中 Kernel

Log 包含核心系統處理封包所呼叫的核心函式紀錄(Function Log)，以及被觸發的核心事件(例如: Link Down/UP 或 IP Changes)的記錄(Event Log)。

接下來我們將本年度計畫的五個工作項目分為三個研究主題分別說明：

主題一：封包導向的核心函式序列追蹤與量測技術之研發及技術工具化

這研究主題主要是追蹤核心系統在處理網路封包時，執行核心函式的時間以及整併所需的其他相關資訊，利用這些資訊我們可以剖析封包在網路協定堆疊(TCP/IP Stack)處理過程中，經過 TCP/IP 各層處理程序以及所花費的時間。

因為 Linux 作業系統中，網路協定堆疊屬於 Kernel Space，Kernel Space 的程式行為都受到核心系統所保護，因此，當我們要輸出 Kernel Space 的系統資訊唯有經系統呼叫(System Call) 或重新修改 Linux 原始碼。但網路協定堆疊之核心函式的數量過於龐大，如果透過系統呼叫輸出訊息，會造成核心系統頻繁地在 Kernel Space 與 User Space 之間作切換，增加核心系統負擔，進而影響本工具的準確性，因此我們採用在 Linux 核心程式嵌入指令的方式來紀錄相關資訊。我們希望插入的指令不會改變原始作業系統的運作模式以及不會影響執行效能。

圖二為核心行為量測系統架構圖，我們先在核心程式插入核心事件擷取與核心函式追蹤的補釘程式碼(Kernel Patches)，當封包經過核心系統的網路協定堆疊或者當核心事件發生時，這些程式碼會將控制權轉移到 Instrument Module 與 Event Module 以精確地追蹤封包經過的核心函式及擷取發生的核心事件。

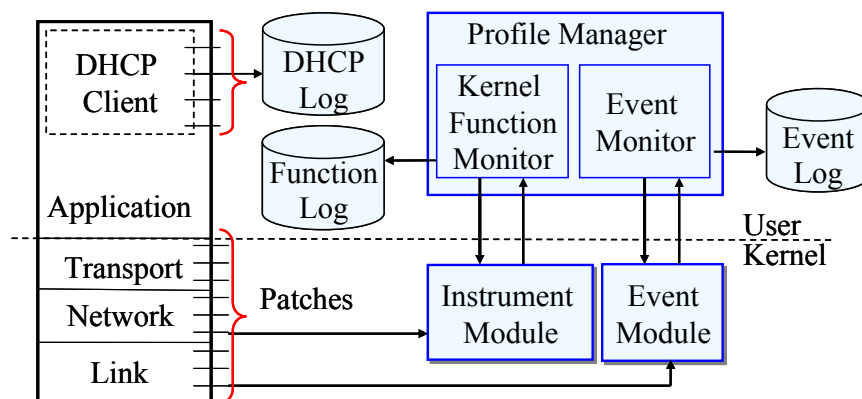
以下分別介紹架構中的模組：

- (a) Linux Kernel Patches: 我們嵌入在核心系統封包處理函式內及核心事件的觸發點，負責轉移控制權的指令（容後說明）。
- (b) Instrument Module: 利用 Linux Kernel Module[2][3]機制所完成，負責紀錄與輸出核心函式執行的記錄，以及管理暫存記錄的記憶體。Instrument Module 可以在不必重新編譯與安裝核心系統程式的情況下，根據 Profile Manager 的指示，動態調整紀錄行為，減少不必要的紀錄動作。
- (c) Event Module: 與 Instrument Module 相似，接受 Probes 移轉控制權，但負責紀錄與輸出核心事件發生的記錄。
- (d) Profile Manager: 負責收集 Instrument Module 產生的 Function Log 以及 Event Module 的 Event Log，此外，也可告知 Instrument Module，使用者指定要記錄

哪些函式。Profile Manager 包含有 Kernel Function Monitor 與 Event Monitor。Kernel Function Monitor 定期將 Instrument Module 內的 Function Log 備份到磁碟的 Log Database 中。Event Monitor 則透過 Netlink 訊息交換機制來記錄 Kernel 中的事件（例如：Router Events 與網路介面 Events）。

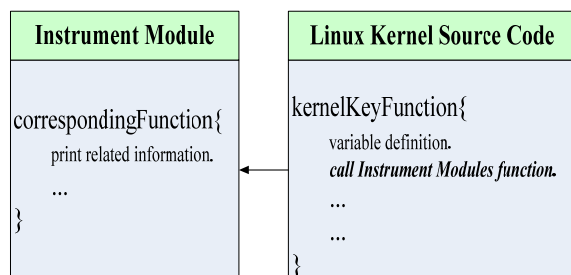
此外，為了進行行動裝置網路交遞(Handover)行為的實驗與分析，本計畫也在 DHCP client 中加入紀錄程式碼來紀錄 DHCP 程式取得 IP 的程序。當 DHCP State Machine 狀態改變時，紀錄程式碼隨即將當時的狀態資訊與系統時間寫入系統日誌檔案中(/var/dhclient.log)。

由圖二的核心行為量測架構圖可看出會影響核心行為的部分，僅包括補釘程式碼、Instrument Module 及 Event Module 三部分。為了減少對系統的影響，我們使用探針的技術來紀錄函式行為。我們在網路協定堆疊的核心函式內，安插一小段程式碼(探針)，這個程式碼只做簡單的判斷與控制權的移轉。如果需要紀錄該函式資訊，探針會將控制權移交給 Instrument Module(如圖三)，由 Instrument Module 的程式碼負責紀錄資訊。反之，則不會作任何事。因此，可藉由 Disable 探針移轉控制權的功能，避免不必要的紀錄動作。



圖二、核心行為量測架構圖

如圖三顯示，探針並不直接輸出資訊，而只是進行控制權的轉移，真正要輸出訊息的程式碼其實是放在 Instrument Module，由 Instrument Module 控制哪些核心函式需要追蹤以及紀錄哪些資訊[4-6]，因此我們可以藉由調整探針移轉控制權的功能，就可避免不必要的紀錄與輸出的動作，因此，不需要為了調整追蹤的核心函式與輸出的資訊，重新編譯與安裝核心系統，增加本工具量測的彈性。

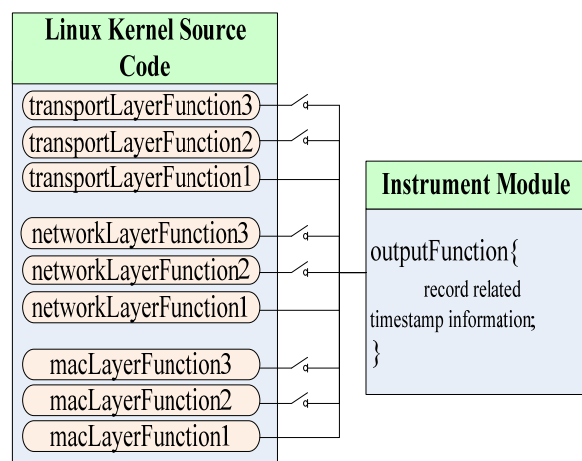


圖三、Linux Kernel Patches 探針示意圖

由於 Instrument Module 與探針同屬於 Kernel Space，所以控制權移交過程中，核心系統不需要浪費較多時間進行 User/Kernel Space 的切換，因此，不會影響核心系統處理網路協定堆疊的運作，而實驗結果也顯示，我們紀錄核心行為的方式對系統效能的影響微乎其微。

此外，我們也提供一個介面讓使用者選擇需要追蹤的核心函式，再由 Kernel Function Monitor 利用 ioctl[7]的機制告知 Instrument Module 哪些核心函式需要追蹤，Instrument Module 根據指示，動態調整探針移轉控制權的功能，追蹤特定函式。

如圖四所示，我們在 Linux 的核心函式設定探針的結果，就好像在核心函式接上一個開關(Switch)連接到 Instrument Module。如果我們要更改想要追蹤的核心函式，可以透過 Kernel Function Monitor 來設定 Instrument Module，阻斷(Switch Off)核心函式與 Instrument Module 之間的連接。



圖四、Instrument Module 決定追蹤函式

另外，我們採用記憶體暫存機制降低記錄行為對系統的影響，同時也利用背景移轉機制解決嵌入式裝置記憶體空間不足的問題。由於核心函式記錄的資訊數量龐大，如果利用常見的 System I/O 方式輸出追蹤記錄，對於核心系統而言是個嚴重的負擔，所以本工具直接使用記憶體記錄追蹤的資訊。

但是，記憶體有可能因為長時間的追蹤過程，而造成記錄空間不足的問題，因此我們設計背景移轉機制。我們利用記憶體映射 (Memory MAP) 機制 [7][8] 將 Instrument Module 內的 Log Buffer 映射到使用者程式的虛擬記憶體分頁(page)，讓在 User Space 的 Kernel Function Monitor 能夠直接讀取 Instrument Module 內的 Log Buffer，定期將 Log Buffer 中的紀錄由記憶體移轉至磁碟中。藉由背景移轉機制，核心系統並不會因為頻繁地系統輸出入 (System I/O)，造成龐大的負擔，也能夠適當地消除記憶體空間不足的問題。

主題二：核心行為記錄與網路協定記錄的整併設計與合成實作

如前所述通訊行為包含核心行為與網路協定行為，因此也利用網路封包擷取工具，擷取網路協定通訊的封包，再利用核心行為的記錄資料，建立內部的核心行為

與外顯的網路協定行為兩者之間的關聯性，利用整併的結果，我們可以合成整體通訊行為。

目前在 Linux 系統環境下，網路封包擷取工具有 tcpdump、Wireshark 等，但是由於這些工具的原始碼過於龐大，分析不易，同時，因為 tcpdump、Wireshark 等網路擷取工具皆使用 libpcap 函式庫撰寫而

成，主要的封包擷取儲存格式也採用的 libpcap 儲存格式。因此，我們選擇利用 libpcap[9][10]撰寫一個可以搭配內部核心行為記錄的網路封包擷取工具，並利用 Source IP, Destination IP, Source MAC, Destination MAC, Identification 與 Packet Length 等 6 項資訊，整合內部的核心行為記錄與外顯的網路協定行為記錄，如圖五。



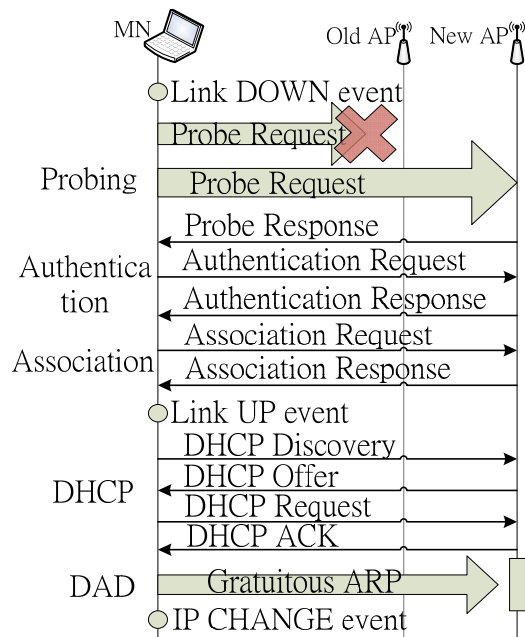
圖五、核心行為記錄與 IP 封包欄位對照表

利用上述工具，我們可以追蹤並整合核心行為與網路協定行為。我們以圖六網路通訊裝置的交遞過程 (Handover Procedure) 為例，當通訊裝置的無線網卡察覺到網路媒介(media)斷線，核心系統即發生 Link DOWN 事件，此時間點表示網路介面已離開原先使用的基地台 (Access Point; AP)，接著，無線網會在各個頻道發出 Probe Request 搜尋可連線的基地台。之後，無線網卡選定目標基地台，對其發出 Authentication Request，等收到目標基地台的 Authentication Response 後，無線網卡

會發出 Association Request 請求連線，等到目標基地台回應 Association Response，則表示連線成功。

通訊裝置的無線網卡重新連上 AP 後，核心系統即發生 Link UP 事件，再利用 DHCP 重新取得 IP。為避免取得的 IP 位址和網路上其他主機的 IP 衝突，核心系統會執行 Duplicate Address Detection (DAD)，DAD 是利用 Gratuitous ARP 詢問其他主機有否使用相同於的 IP 位址。確定新取得的 IP 位址沒有其他主機在使用，核心系統會進行網路設定，等到核心系統發

生 IP CHANGE 事件時，則表示整個交遞過程完成。



圖六、網路通訊裝置之交遞過程

主題三：整合測試與評比規劃

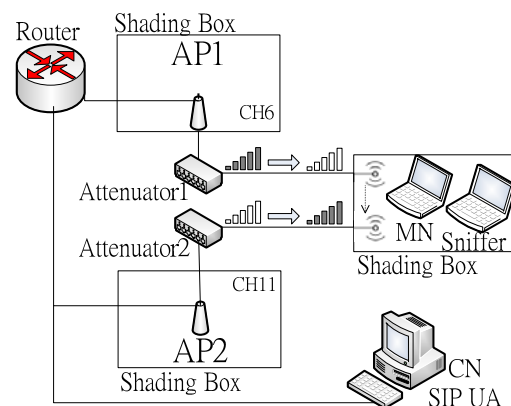
經過封包導向的核心函式序列追蹤與量測的技術之研發後，我們已經具備核心行為擷取與追蹤技術，接著須建構實際的測試例，以展示我們的工具測試整體通訊行為(含核心函式、核心事件與網路協定行為)的功能。

如前所述，網路通訊裝置在交遞過程中，除了執行核心函式外，也會觸發一些核心事件(例如: Link UP、Link DOWN、IP CHANGE 等等)，而一般封包傳輸的過程較少產生核心事件，所以我們使用 SIP-based VoIP 軟體建構測試例，以展示我們的工具在行動主機漫遊時，可以記錄處理封包的核心函式以及觸發的核心事件。配合外顯的網路協定行為的記錄，我們就可剖析網路通訊裝置的交遞過程的整體通訊行為。

圖七說明我們建置的無線網路測試環境，為了避免外界訊號干擾影響實驗結

果，我們使用訊號阻隔箱(Shading Box)和可程式化衰減器(Attenuator)建置實驗環境，我們將通訊裝置、封包擷取主機與兩個基地台的天線放在同一個阻隔箱，基地台的天線訊號先經由可程式化衰減器，再連接到阻隔箱內的天線。如此，我們就可藉由可程式化衰減器控制天線訊號，模擬出通訊裝置漫遊交遞的情境。

在通訊過程中，觀察 MN 移動發生交遞時，核心事件觸發與擷取的情況。所以在通訊過程中，讓 MN 從原服務基地台 (AP1) 的訊號涵蓋範圍移動至目標基地台 (AP2)。當行動節點離開 AP1 範圍時，會觸發 Link DOWN；之後進入 AP2 範圍，會觸發 Link UP。



圖七、無線網路測試環境圖

圖八是本測試例實測過程產生的核心事件、核心函式以及封包之紀錄，經由 Integrated Analyzer 整併與顯示的結果。Integrated Analyzer 的 GUI 介面共分為三大部份: Packet Viewer、Kernel Function Viewer 和 Kernel Event Viewer。Packet Viewer 主要目的在於瀏覽封包擷取器所擷取到的網路封包，Kernel Function Viewer 用來顯示處理封包的核心函式與執行時間點。Kernel Event Viewer 則顯示被觸發的核心事件與時間點。目前我們已建立網路

封包與核心函式以及核心事件紀錄的關連性，在 Packet Viewer 選取某一網路封包，就可以在 Kernel Function Viewer 與 Kernel Event Viewer 顯示該封包相對應的核心函式和核心事件的紀錄。未來我們會進一步

研發 Integrated Analyzer 的過濾(Filtering)與合成(Synthesis)，可以協助使用者過濾不必要的資訊、合成或推導使用者關心的網路通訊行為，以及展示效能測量與分析結果。

No	Time	Source	Destination	Protocol	Info		
2	17:21:25.460...	60.60.1.254	239.255.255.250	UDP	Ident: 0000 S-Port: 3105 (Other), D-Port: 1900 (Other) Len: 320	} Sniffer Log	
3	17:21:25.505...	60.60.1.254	60.60.1.66	ARP	Type: REQ S-MAC: 00:12:CF:38:12:3A Target: 00:00:00:00:00:00		
17:21:25.505542 netif_rx_Func_Start						} Kernel Function Log	
17:21:25.505546 netif_receive_skb_Func_Start							
No	Time	Event	Description				
1	09:47:02.296074	NETWORK	wlan0 State: Link DOWN				} Kernel Event Log
2	09:47:14.298850	ROUTING	Deleted 60.60.1.0/24 dev wlan0 src 60.60.1.65				
3	09:47:14.298932	ROUTING	Deleted default via 60.60.1.254 dev wlan0				
4	09:47:14.330944	NETWORK	Deleted wlan0 inet 60.60.1.65				
5	09:47:14.332786	ROUTING	Deleted broadcast 60.60.1.255 dev wlan0 src 60.60.1.65				
6	09:47:14.332857	ROUTING	Deleted broadcast 60.60.1.0 dev wlan0 src 60.60.1.65				
7	09:47:14.332895	ROUTING	Deleted local 60.60.1.65 dev wlan0 src 60.60.1.65				
8	09:47:14.333535	NETWORK	Deleted wlan0 inet fe80::21f:e1ff:fe65:fe24				
11 09:47:43.119497 NETWORK wlan0 State: Link UP						} Kernel Event Log	
...							
17 09:47:45.051001 NETWORK Assign wlan0 inet fe80::21f:e1ff:fe65:fe24							
18 09:47:47.003967 DHCPCLIENT DHCPDISCOVER on wlan0 to 255.255.255.255 port 67 interval 3							
19 09:47:47.021638 DHCPCLIENT DHCPPOFFER from 30.30.1.254							
20 09:47:47.021712 DHCPCLIENT DHCPREQUEST on wlan0 to 255.255.255.255 port 67							
21 09:47:47.045599 DHCPCLIENT DHCPACK from 30.30.1.254							
...							
28 09:47:47.091010 NETWORK Assign wlan0 inet 30.30.1.66						} Kernel Event Log	
29 09:47:47.091447 ROUTING local 30.30.1.66 dev wlan0 src 30.30.1.66							
30 09:47:47.092294 ROUTING broadcast 30.30.1.255 dev wlan0 src 30.30.1.66							
31 09:47:47.092323 ROUTING 30.30.1.0/24 dev wlan0 src 30.30.1.66							
32 09:47:47.092344 ROUTING broadcast 30.30.1.0 dev wlan0 src 30.30.1.66							
33 09:47:48.096700 ROUTING default via 30.30.1.254 dev wlan0							

圖八、測試例實測結果紀錄整併結果

■ VoIP Handover 行為分析

利用本工具我們可以歸納 Handover 發生時，斷線與再連過程的各個階段的延遲時間。因此，我們也設計無線 VoIP 的漫遊交遞實驗，展示如何利用本工具來分析在 VoIP 的 Handover 過程中，Fedora Linux 的網路通訊行為。

我們從配備 Atheros driver 的 Fedora Linux 行動裝置撥打 VoIP 電話(Skype)至對應主機(Corresponding Node)，於通話 60 秒後 關掉 AP1 的電源，此時可以觀察到行動裝置會 Handover 至 AP2。透過本工具可以擷取並整併 Handover 過程的 Fedora Linux 核心函式、核心事件以及網路封包的紀錄，利用這些紀錄，我們可以觀察的 Handover 行為以及計算 Handover 過程中各個階段的時間延遲。

圖九展示我們測量的結果，從結果來

看，Fedora Linux 的 Network Manager 與 Atheros driver 處理 Handover 的表現不好，效能上有很大的改善空間。當行動裝置與 AP 失聯後，Fedora Linux 會優先嘗試與前一次連線成功的 AP 進行連線。這部份花了非常多的時間，跟 Windows PC 比較起來，差異非常的大，因為 Windows PC 能夠在 3 秒內就開始嘗試與其它的 AP 進行網路連線。

Phases	Times
L2H Detection Method Delay	27.628691s
Probing and Probe Delay	6.787582s
Authentication Delay	1.683394s
Association Delay	0.00206s
L3H Detection Method Delay	3.87256s
DHCP Delay	0.035146s
DAD Delay	N/A

圖九、VoIP Handover 延遲測量結果

從 Probing and Probe Delay 的結果來看，Atheros driver 花了將近 7 秒鐘來判斷出 Link 斷掉了，之後再通知 Linux 核心 Link Down 已經發生。

從 L3H Detection Method Delay 的結果來看，Linux OS 在收到 Association Response 封包後，並沒有立刻呼叫 DHCP Client 協議，有將近 4 秒鐘的延遲，這部份可能得檢查 Linux Network Manager 原始碼才能得知在這 4 秒鐘之內，Linux 在做什麼事情。此外這版本的 DHCP Client 也不會進行 Duplicate Address Detection (DAD)的動作。

四、結果與討論

我們已完成本年度的工作目標，初步驗證，我們確實可以記錄核心行為，並將核心行為紀錄與協定行為紀錄整併，幫助使用者了解整體通訊行為。

為了進一步分析我們系統的效能，我們也規劃 UDP 與 TCP 兩種傳輸方式，以測試我們的系統效能，了解我們的工具處理網路封包的效能。

UDP 傳輸測試是以 Iperf 產生 UDP 封包，測試我們的系統在不同封包流量下，封包遺失的情形。我們由一台 PC 以 5Mbps 至 1000Mbps 多種不同的速度，傳送 UDP 封包給另一台 PC，每一次實驗 30 秒鐘。由兩部 PC 上的記錄檔，我們可以得知不同傳輸速度的封包遺失率。

圖十展示 Iperf 量測的結果，如圖所示，即便在待測裝置上同時執行量測系統與封包擷取程式(Profiler+Sniffer)，傳輸速度在 50Mbps 以內，封包完全不會有遺失的情形。

除了封包遺失外，我們也以 FTP 傳輸產生 TCP 封包以測量本工具的傳輸效率，圖十一展示我們測量的結果。

Speed ↓ (Mbps)	Patched Linux kernel			Original ↑ Linux kernel
	Profiler + SNIFFER	Profiler only	None	Loss Rate (%)
	Loss Rate (%)	Loss Rate (%)	Loss Rate (%)	
5	0	0	0	0
50	0	0	0	0
100	0.32	0.13	0.086	0.086
200	44	0.91	0.83	0.34
400	82	68	18	0.37
500	93	89	64	0.45
1000	99	99	99	0.55

圖十、Iperf 網路效能測試結果

Wire FTP Test								
File ↓ Size ↓ (MB)	Linux kernel with patch						Original Linux kernel	
	Profiler + SNIFFER		Profiler only		None		None	
	Time ↓ (S)	Speed ↓ (MB)	Time ↓ (S)	Speed ↓ (MB)	Time ↓ (S)	Speed ↓ (MB)	Time ↓ (S)	Speed ↓ (MB)
5	0.23	23	0.21	24	0.21	24	0.18	29
10	0.41	26	0.39	26	0.39	26	0.38	27
50	2.02	25	1.98	26	1.96	26	2.1	26
100	4.01	25	3.93	26	3.92	26	3.89	26

圖十一、FTP 測試結果

如圖十一 FTP 網路實驗的結果顯示，在下載 50MB 以下檔案大小的時候，我們系統與原來系統的效能差不多(有時還比較好，但這應該是實驗的誤差，我們需要更多的實驗來驗證)。但在 50MB 到 100MB 之間，速度開始下滑。另外從 IPerf 實驗結果得知，當傳輸速度大於 50Mbps 以後，開始出現封包遺失的現象。

從實驗結果看來，以本工具來評量傳輸速度 50Mbps 以下的裝置，並不會有太大的問題的。本系統的目的是在協助通訊行為的除錯與分析，因此，著重在裝置之網路連線與通訊協定(例如：鏈結層連線、DHCP、IP 與路由表的改變以及網路協定封包的處理等。)的行為紀錄與分析，而非一般資料 (Data) 傳輸的效能量測，因此傳輸速度不是本工具的設計考量，而且 50Mbps 的傳輸速度已遠超過一般裝置的傳輸需求。

以上實驗驗證我們的追蹤與量測工具已經可以用來整合(1)核心內部通訊協定堆疊之分析、(2)核心內部網路事件通知機制和(3)封包擷取工具，已達成本年度計畫目標。未來我們預期研發一套具有圖形化介面的整合分析工具，呈現通訊行為之因果關係的分析結果，協助使用者深入瞭解嵌入式網路通訊裝置之網路協定行為，並且界定出網路協定行為的效能瓶頸，此外，為了因應核心系統的版本更新，我們也預計研發一套解譯核心系統並自動嵌入追蹤指令的機制。

五、 結論

隨著網路與電腦科技的進步，嵌入式網路通訊裝置已經逐漸普遍於市面上，然而目前對於嵌入式網路通訊裝置的相關效能評估工具仍不多，使得我們有必要開發一套合適的輔助工具，協助嵌入式網路通訊裝置的開發者診斷通訊行為的缺失，界定通訊行為的效能瓶頸，進而分析問題及改善缺失(例如縮短延遲、加快反應時間及減少封包遺漏等)。

本計畫所提出「嵌入式網路通訊裝置之核心行為與網路協定行為整合測試、分析與評比工具」便是希望使用者可以經由相關網路協定的測試，深入瞭解嵌入式網路通訊裝置在通訊行為過程中所發生內部的核心行為，以及外顯的網路協定行為，並且藉由這些測試的資料記錄，可以用來分析網路封包在核心系統內部網路各層的處理程序、核心事件以及執行或延遲時間，進一步分析整體通訊行為的缺失或效能。

未來工作與研究

我們的終極目標是希望可以發展一套嵌入式網路通訊裝置之核心行為與網路協

定行為整合測試、分析與評比工具，本年度我們已整合(1)核心內部網路協定堆疊之分析、(2)核心內部網路事件通知機制以及(3)封包擷取工具，並初步驗證我們的工具確實可整併系統內部核心行為與外顯的網路協定行為。

然而，在核心內部網路協定堆疊之分析部分，我們本年度以人工方式，找出數十個在網路協定堆疊中具有代表性的核心函式，只追蹤網路封包在這數十個核心函式傳輸過程，未來我們會研發一套解譯核心系統並自動嵌入追蹤指令的機制，自動找出與網路封包相關的核心函式，如此，不僅能夠更精確的追蹤核心函式，同時也可更方便將我們的工具移植到新版本的核系統。在核心內部網路事件通知機制部分，我們也會找出更多重要且有意義的核心事件，讓使用者更加深入瞭解嵌入式網路通訊裝置之通訊行為。

最後是智慧型的診斷分析，上述的整合分析工具已可以整併系統內部與外顯的整體通訊行為，但仍需要更多的過慮、分析與解譯的功能，因此，我們也要發展一套智慧型診斷機制，針對這些綜合資訊作進一步的分析，提供更直接、更精確的資訊，協助使用者更精準界定問題或效能瓶頸。以上都是我們未來所要研究的課題。

計畫成果自評

本計畫依時程順利進行，設計與實作完成核心事件擷取與核心函式追蹤的追蹤與量測技術，以及一套可以整併核心函式、核心事件與網路封包的整合分析工具的先期系統雛型，並發表多篇論文，提出多項專利申請，培養碩博士研究生多位，衍生多個產學計畫，目前也有一項技術移轉進行中，成果豐碩。

參考文獻

- [1] Chuanxiong Guo and Shaoren Zheng, "Analysis and Evaluation of the TCP/IP Protocol Stack of Linux," Proc. International Conference on Communication Technology, vol 1, pp. 444–453, Aug. 2000.
- [2] Bryan Henderson, "Linux Loadable Kernel Module HOWTO," Available from: <http://www.tldp.org/HOWTO/Module-HOWTO/>
- [3] Jongmoo Choi, "Kernel aware module verification for robust reconfigurable operating system," Journal of Information Science and Engineering, Vol. 23 No. 5, pp. 1339-1347, Sep. 2007.
- [4] R. J. Moore, "Dynamic probes and generalized kernel hooks interface for Linux," Proc. 4th Annual Linux Showcase and Conference, pp. 139-145, 2000.
- [5] Nicolas Lorient and Jean-Marc MenaudGeneralized, "Dynamic Probes for the Linux Kernel and Applications with Arachne," Proc. 2007 IADIS Conference on Applied Computing, Feb. 2007.
- [6] R. Krishnakumar, "Kernel Korner: Kprobes—a Kernel Debugger," Linux Journal, Jun. 2006.
- [7] Jonathan Corbet, Alessandro Rubini, Greg Kroah-Hartman, "Linux Device Driver – 3rd Edition," O'Reilly, 2005
- [8] John Shapley Gray, "Interprocess communication in Linux", Pearson Education Taiwan Ltd.
- [9] Tim Carst, "Tutorial: Programming with pcap", Available from: <http://www.tcpdump.org/pcap.htm>
- [10] Wireshark Wiki – "LibpcapFileFormat", Available from: <http://wiki.wireshark.org/Development/LibpcapFileFormat>
- [11] Kai-Chen Hsu, "Design and Implementation of a Middleware for Network-Aware Applications," Master Thesis, NCTU, 2006.
- [12] Yuan-Li Hsu, "Analysis of Link-layer and Cross-layer Handover Behavior of IEEE 802.11 Wireless Networks," Master Thesis, NCTU, 2009
- [13] Steve Best, "Linux® Debugging and Performance Tuning: Tips and Techniques," Prentice Hall, Oct. 2005.
- [14] Mireille Ducasse and Jacques Noye, "Tracing Prolog programs by source instrumentation is efficient enough," The Journal of Logic Programming, pp. 157–172, 2000.
- [15] Nikolai Joukov, Avishay Traeger, Rakesh Iyer, Charles P. Wright, and Erez Zadok, "Operating System Profiling via Latency Analysis," Proc. 7th symposium on Operating Systems Design and Implementation.
- [16] Yao Guo, Ziwen Chen, Xiangqun Chen, "A Lightweight Dynamic Performance Monitoring Framework for embedded Systems," Proc. International Conference on Embedded Software and Systems, 2009.
- [17] Abhinav Bhatele and Guojing Cong, "A Selective Profiling Tool: Towards Automatic Performance Tuning," Proc. Parallel and Distributed Processing Symposium, 2007.
- [18] Nicholas Mc Guire, "Kernel Function Instrumentation – KFT," 2006.
- [19] Prashanth P. Bungale and Chi-Keung Luk, "PinOS: A Programmable

- Framework for Whole-System Dynamic Instrumentation,” Proc. 3rd International Conference on Virtual Execution Environments.
- [20] Luiz DeRose and Ted Hoover Jr., “The Dynamic Probe Class Library - An Infrastructure for Developing Instrumentation for Performance Tools,” Proc. 15th International Parallel and Distributed Processing Symposium.
- [21] Kiduk Kwon, Midori Sugaya, and Tatsuo Nakajima, “Analysis of Embedded Linux using Kernel Analysis System,” Proc. International Conference on Embedded Software and Systems, 2009.
- [22] Mathieu Desnoyers and Michel R. Dagenais, “Low disturbance embedded system tracing with Linux Trace Toolkit Next Generation,” Proc. Embedded Linux Conference, 2006.
- [23] Mathieu Desnoyers and Michel R. Dagenais, “LTTng, Filling the Gap Between Kernel Instrumentation and a Widely Usable Kernel Tracer,” Linux Foundation Collaboration Summit 2009.
- [24] Karim Yaghmour and Michel R. Dagenais, “Measuring and characterizing system behavior using kernel-level event logging,” Proc. USENIX Annual Technical Conference, 2000.
- [25] Mathieu Desnoyers and Michel R. Dagenais, “The LTTng tracer A low impact performance and behavior monitor for GNU Linux,” Proc. Ottawa Linux Symposium (OLS), 2006.
- [26] P Heidari, M Desnoyers, and M Dagenais, “Performance Analysis of Virtual Machines Through Tracing,” Proc. Canadian Conference on Electrical and Computer Engineering, 2008.
- [27] Tim Bird, “Learning the kernel and finding performance problems with kfi,” CELF International Technical Conference, 2005.
- [28] Aroon Nataraj, Allen D. Malony, Sameer Shende, and Alan Morris, “Kernel-Level Measurement for Integrated Parallel Performance Views the KTAU Project,” Proc. IEEE International Conference on Cluster Computing, 2006.
- [29] Ananth Mavinakayanahalli, Prasanna Panchamukhi, Jim Keniston, Anil Keshavamurthy, and Masami Hiramatsu, “Probing the Guts of Kprobes,” Proc. Ottawa Linux Symposium (OLS), 2006.
- [30] Byungjoon Lee, Seong Moon, and Youngseok Lee, “Application-Specific Packet Capturing using Kernel Probes,” Proc. 11th IFIP/IEEE international conference on Symposium on Integrated Network Management, 2009.
- [31] Vara Prasad, Jim Keniston, William Cohen, Frank Ch. Eigler, Martin Hunt, and Brad Chen, “Locating System Problems Using Dynamic Instrumentation,” Proc. Ottawa Linux Symposium (OLS), July 2005.
- [32] Mike Mason, “Using SystemTap for Dynamic Tracing and Performance Analysis,” Proc. Ottawa Linux

Symposium (OLS), 2007.

- [33] Frank Ch. Eigler, “Problem Solving With Systemtap,” Red Hat Summit 2007.
- [34] Ariel Tamches and Barton P. Miller, “Fine-grained dynamic instrumentation of commodity operating system kernels,” Proc. 3rd symposium on Operating systems design and implementation.
- [35] A.Tamches, and B.P.Miller., “Using Dynamic Kernel Instrumentation for Kernel and Application Tuning,” Proc. International Journal of High Performance Computing Applications, 1999.
- [36] Y. Yanagisawa, K. Kourai, and S. Chiba, “A dynamic aspect-oriented system for OS kernels,” Proc. 5th international Conference on Generative Programming and Component Engineering, 2006.
- [37] Bryan M. Cantrill, Michael W. Shapiro and Adam H. Leventhal, “Dynamic Instrumentation of Production Systems,” Proc. USENIX Annual Technical Conference, 2004.
- [38] WE Cohen, “Tuning programs with OProfile,” Wide Open Magazine, 2004

附錄 A、著作成果

Journal Papers:

1. Kuang-Hui Chi, Chien-Chao Tseng and Ya-Hsuan Tsai. "Fast Handoff among IEEE 802.11r Mobility Domains," *Journal of Information Science and Engineering*, Vol. 26, No. 4, pp. 1345-1362, 2010.
2. Yu-Wen Chen, Jui-Tang Wang, Kuang-Hui Chi and Chien-Chao Tseng. "Group-Based Authentication and Key Agreement," (accepted and to appear in) *Wireless Personal Communications*.
3. Kuang-Hui Chi, Yung-Chien Shih, Ho-Han Liu, Jui-Tang Wang, Shiao-Li Tsao and Chien-Chao Tseng. "Fast Handoff in Secure IEEE 802.11s Mesh Networks," " (accepted and to appear in) *IEEE Transactions on Vehicular Technology*.

Conference Papers:

1. Kuei-Li Huang, Jui-Tang Wang, Jen-Shun Yang, and Chien-Chao Tseng. "Radio Resource Management Scheme for Multicast/Broadcast Services in WiMAX Networks," *Mobile Computing Workshop*, May 2010.
2. Kuei-Li Huang, Jui-Tang Wang, Jen-Shun Yang, and Chien-Chao Tseng. "A Boundary Detection Scheme for Multicast/Broadcast Service Zones in WiMAX Networks" *Mobile Computing Workshop*, May 2010.
3. Yuan-Chun Chien, Chien-Chao Tseng, Kai-Cheng Hsu and Jack Tu. "Interface Management and Network Selection for Mobile Devices with Multiple Adaptors," 數位生活科技研討會, 2010.

Patents:

1. Jun-Yu Lai, Kue-Li Huang, Jui-Tang, Wang, Chien-Chao Tseng and Chao-Nan Wu. "Routing Method and Routing Path Recovery Mechanism in Wireless Sensor Network Environment (無線感測網路環境中之繞送方法及繞送路徑之修復方法)," ROC and USA Patent Pending.
2. Chien-Chao Tseng, Yung-Sheng Lee, Yu-Jen Chang and Elia Chen. "Network Association Method For Public Transportation System (應用於大眾運輸系統中的網路連結建立方法)," ROC Patent Pending.
3. Kue-Li Huang, Chao-Nan Wu, Jui-Tang, Wang and Chien-Chao Tseng. "無線感測網路中具省電效果之骨幹導向結構、建立方法及修復方法," " ROC and USA Patent Pending.
4. Chien-Chao Tseng, Tsung-Hung Li and Hung-Hsin Chang, "A Mechanism for Tracing Network Packet Processing Procedure (一種追蹤網路封包處理程序的機制)," ROC Patent Pending.

國科會補助計畫衍生研發成果推廣資料表

日期 2010年09月13日

國科會補助計畫	計畫名稱: 子計畫三: 嵌入式網路通訊裝置核心與網路協定行為效能分析評比技術與工具之研發(中心分項)(1/2) 計畫主持人: 曾建超 計畫編號: 98 -2220-E -009 -047 - 學門領域: 自由軟體暨嵌入式系統		
研發成果名稱	(中文) 一種追蹤網路封包處理程序的機制		
	(英文) Method for Tracing Processing Procedures of Network Packets		
成果歸屬機構	國立交通大學	發明人 (創作人)	曾建超, 李宗鴻, 張弘鑫
技術說明	(中文) 本發明提出一種追蹤網路封包處理程序的機制，可藉由核心函式存取網路封包資料，追蹤網路封包在核心系統的處理程序。應用在網路通訊裝置中，該機制可以利用網路封包的資料結構，找出存取封包的函式，並在該函式嵌入指令，記錄執行函式與封包內容，追蹤系統內部處理網路封包的行為與時間點，協助開發者分析系統內部的網路行為，釐清與改善網路通訊問題。本發明包含一個函式分析方法，該方法可以找出系統中有存取網路封包資料的函式，並在這些函式中嵌入指令，藉由程式本身執行函式的順序，依序觸發嵌入指令來記錄被執行的函式名稱或代碼。		
	(英文) This invention presents a new mechanism that can trace the execution sequence of kernel functions that process network packet. With such trace, we can analyze the networking behavior, conduct software debugging and optimize performance of networking devices. The underlying idea of the mechanism is that the execution sequence of functions process a network packet can be derived from the sequence the functions access the data structure of the network packet. The proposed mechanism first adopts a function analyzer to identify all functions that refer or handle the data structure of network packets, and then patches instructions in each function identified. At run time, the execution of each patched function will trigger the patched instructions automatically to record the function identity and other		
產業別	資訊服務業		
技術/產品應用範圍	網通產品開發、測試與應用		
技術移轉可行性及預期效益	已可移轉、預期效益高		

註：本項研發成果若尚未申請專利，請勿揭露可申請專利之主要內容。

98 年度專題研究計畫研究成果彙整表

計畫主持人：曾建超			計畫編號：98-2220-E-009-047-				
計畫名稱：嵌入式網路通訊裝置評比技術與工具之研發--子計畫三：嵌入式網路通訊裝置核心與網路協定行為效能分析評比技術與工具之研發(中心分項)(1/2)							
成果項目			量化			單位	備註（質化說明：如數個計畫共同成果、成果列為該期刊之封面故事...等）
			實際已達成數（被接受或已發表）	預期總達成數(含實際已達成數)	本計畫實際貢獻百分比		
國內	論文著作	期刊論文	1	0	50%	篇	
		研究報告/技術報告	0	0	100%		
		研討會論文	0	0	100%		
		專書	0	0	100%		
	專利	申請中件數	1	0	100%	件	
		已獲得件數	0	0	100%		
	技術移轉	件數	0	1	100%	件	
		權利金	0	0	100%	千元	
	參與計畫人力（本國籍）	碩士生	3	0	100%	人次	
		博士生	1	0	100%		
		博士後研究員	0	0	100%		
		專任助理	0	0	100%		
國外	論文著作	期刊論文	2	0	70%	篇	
		研究報告/技術報告	0	0	100%		
		研討會論文	0	0	100%		
		專書	0	0	100%	章/本	
	專利	申請中件數	0	0	100%	件	
		已獲得件數	0	0	100%		
	技術移轉	件數	0	0	100%	件	
		權利金	0	0	100%	千元	
	參與計畫人力（外國籍）	碩士生	0	0	100%	人次	
		博士生	0	0	100%		
		博士後研究員	0	0	100%		
		專任助理	0	0	100%		

其他成果 (無法以量化表達之成果如辦理學術活動、獲得獎項、重要國際合作、研究成果國際影響力及其他協助產業技術發展之具體效益事項等，請以文字敘述填列。)	無
---	---

	成果項目	量化	名稱或內容性質簡述
科 教 處 計 畫 加 填 項 目	測驗工具(含質性與量性)	0	
	課程/模組	0	
	電腦及網路系統或工具	0	
	教材	0	
	舉辦之活動/競賽	0	
	研討會/工作坊	0	
	電子報、網站	0	
	計畫成果推廣之參與（閱聽）人數	0	

國科會補助專題研究計畫成果報告自評表

請就研究內容與原計畫相符程度、達成預期目標情況、研究成果之學術或應用價值（簡要敘述成果所代表之意義、價值、影響或進一步發展之可能性）、是否適合在學術期刊發表或申請專利、主要發現或其他有關價值等，作一綜合評估。

1. 請就研究內容與原計畫相符程度、達成預期目標情況作一綜合評估

☒ 達成目標

☐ 未達成目標（請說明，以 100 字為限）

☐ 實驗失敗

☐ 因故實驗中斷

☐ 其他原因

說明：

2. 研究成果在學術期刊發表或申請專利等情形：

論文：☐ 已發表 ☐ 未發表之文稿 ☒ 撰寫中 ☐ 無

專利：☐ 已獲得 ☒ 申請中 ☐ 無

技轉：☐ 已技轉 ☒ 洽談中 ☐ 無

其他：（以 100 字為限）

3. 請依學術成就、技術創新、社會影響等方面，評估研究成果之學術或應用價值（簡要敘述成果所代表之意義、價值、影響或進一步發展之可能性）（以 500 字為限）

狹倉??瞄陌倭鮎腔猾婦懽研昭瞄陌滲宅陡蹈祓?鷄講習撮胍

狹倉瞄陌倭鮎??鷄鋒繚?隅??腔涇罔乙盱摯料?倭鮎(漪瞄陌滲宅陡蹈}瞄陌岬璃鷄鋒繚
猾婦)磁倉腔傑?鷄?鈎〔

襁盱?狗ズ宅規离嶠追氦僅咭料?倭鮎腔[≡]囧麼、蝨乙窺傻旃追筍、〔

羶蛭齟鋒料規离腔嶠追乙礬齟 鋒料家枯鈕嫫湮耆珣〔

醴ゴ店?湮斐嶠追蚊蛭氦煦昂馱檢乙源晞蚊蛭氦蚊蛭掛炆緋〔