Linux 路由器：建構、測試與追蹤

Ⅰ. 實驗目的

本實驗中將學習如何使用一般個人電腦建構一台 Linux 路由器，再分別使用除錯工具軟體與測試方法，了解此路由器之設定及運作方式。

實驗一開始先學習使用 Linnux 路由 (route) 的基本指令以及了解其設定方式，接著安裝一套 routing daemon 套件－Quagga，並學習其設定方式。最後安裝 KDB 套件，追蹤核心內對於 routing 的處理流程。

Ⅱ. 實驗設備

一、硬體

項目	數量	備註
個人電腦	1	本身需具有 on board 的網路卡。
網路卡	1	需選用 Linux kernel 有支援之網路卡。

二、軟體

項目	版本	數量	備註
Linux distribution	Fedora V [1]	1	本實驗採用 Fedora 系統的 Linux 來說明。採用其他 distribution 亦可。Linux 可由各大 FTP 站取得。
Linux kernel[2]	2.6.16	1	可以到 http://www.kernel.org/取回。因爲我們選的 KDB patch 版本的關係，所以我們選用此一版本。
Kdb[3]	4.4	1	至 http://oss.sgi.com/projects/kdb/下載此軟體。此軟體爲一 kernel patch 檔、需與 kernel 版本互相搭配。
Quagga[4]	0.99.4-1	1	請至 http://www.quagga.net/下此軟體。

Ⅲ. 背景資料

一、路由器概念

一般來說，電腦數量小於數十部的區域網路不需要路由器，只需要用 hub 或 switch 連接每一部電腦， 然後透過單一線路連接到 Internet。但如果是電腦數量過多的網路環境， 就會需要考量到實際佈線的困難以及效能。例如大樓內不同樓層要使用 hub/switch 連接所有的電腦，在佈線上相當困難。要解決這個問題，可以透過每一個樓層架設一部路由器， 並在各樓層間，用路由器相連接，就能夠簡單的管理各樓層的網路；否則，因爲各樓層之間沒有架設路由器，而是直接以網路線串接各樓層的 hub/switch 時，由於同一網域的資料是透過廣播來傳遞的，整棟大樓的電腦因而處於同一 collision domain，當整個大樓的某一部電腦在廣播時， 所有的電腦將會予以回應，會造成大樓內網路效能低落。所以架設路由器將實體線路區隔開，能夠區隔出各樓層之間的 collision domain，藉以提昇網路效能。

由於各樓層之間爲不同網域，當主機想要將資料傳送到不同的網域時便得透過路由器。路由器會分析來源端封包的 IP 表頭，找出目標的 IP 後，透過路由器本身的路由表 (routing table) 將這個封包向下一個目標傳送。

二、Linux Route 介紹

通常，在 Linux 系統上的路由都是靜態路由，也就是由系統管理員使用”route”命令所加入之靜態路由規則。

相對於靜態路由，另一種路由方式爲動態路由。動態路由的規則是由各路由器之間，藉由路由協定程式互相交換路由規則而形成的。常見的路由協定有 RIPv1、RIPv2、ISIS、BGP 等。

以下介紹幾個在 linux 中處理網路封包路由之函式：

ip_rt_ioctl：

此函式主要負責處理使用者以 route 指令所加入或刪除之靜態路由規則。如果要作的是刪除路由的動作，則先清除路由表中的規則，再檢查快取中有無副本，有的話也一併刪除，以確保其一致性。如果請求是增加路由的話，則先檢查指定的介面是否已存在此路由規則，沒有的話便新增這個路由規則，但此時並不會在路由快取新增資料。

ip_route_input：

每次的進入系統的封包，都會觸發此函式，查詢此封包之路由。

首先會使用 hash function rt_hash_code()來查詢此封包之路由資訊，是否

存在系統之 route cache，若不存在，則接著觸發 ip_route_inpute_mc 對此封包作處理。

ip_route_input_mc：

當發現該封包之目的位址爲 multicast，則將此封包用此函式處理，否則就交給函式 ip_route_input_slow 處理。

ip_route_input_slow：

當封包之路由資訊不存在於系統之快取且不屬於 multicast 的封包，則會由此函式作處理。

三、Linux 之 route 指令

在 Linux 系統中，使用"route –n"，可以列出目前系統中的路由規則。

```
[root@localhost /]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth0
```

欄位名稱	功能
Kernel IP routing table	Kernel 內的 IP routing table
Destination	目的地的 IP 網路位址（Network Address）
Gateway	Gateway 的 IP address
Genmask	Destination 的 subnet mask
Flag	用來指示此 route rule 的狀態
Metric	需要經過幾個 hops 才能到達 destination
Ref	Reference 到此 rule 之 daemon 個數，如 RIP
Use	至目前爲止，使用此 rule 之封包個數
Iface	此 rule 所套用之網路介面(如 eth0、eth1)

四、追蹤 Linux 核心

本實驗採用 KDB 來追蹤 Linux 核心。目前在 Linux 上常用的 GDB，其對 kernel 除錯功能僅限於讀取核心資料而不能使用如設定 breakpoints 或者 step by step 的執行。其它的延伸套件(如 kdebug)則多了一些優點，包括對模組進行除錯，但仍缺乏以上二項功能。

KDB 提供在本機端設定 breakpoints 以及 step by step 執行的動作。GDB

雖可達到相同目的，但卻需要使用者透過 serial port 在一台機器上執行而對另外一台機器進行除錯。

KDB 的指令列表如下：

Command	Description
bc	Clear Breakpoint
bd	Disable Breakpoint
be	Enable Breakpoint
bl	Display breakpoints
bp	Set or Display breakpoint
bpa	Set or Display breakpoint globally
bt	Stack Traceback
btp	Display stack for process <pid>
cpu	Switch cpus
env	Show environment
ef	Display exception frame
go	Restart execution
help	Display help message
id	Disassemble Instructions
ll	Follow Linked Lists
md	Display memory contents
mds	Display memory contents symbolically
mm	Modify memory contents
ps	Display active task list
reboot	Reboot the machine
rd	Display register contents
rm	Modify register contents
sr	Magic SysRq key
ss	Single Step
ssb	Single step to branch/call
set	Add/change environment variable

本實驗需要在 kernel 設定中斷點，觀察在 stack 中的資料及執行單個指令的功能。以下爲節錄自其 manual 的相關指令說明及其範例。

1. bp 和 bd

bp schedule
Sets an instruction breakpoint at the begining of the function schedule.
bp schedule+0x12e
Sets an instruction breakpoint at the instruction located at schedule+0x12e.
bp ttybuffer+0x24 dataw
Sets a data write breakpoint at the location referenced by ttybuffer+0x24 for a length of four bytes.
bp 0xc0254010 datar 1
Establishes a data reference breakpoint at address 0xc0254010 for a length of one byte.
bp List current breakpoint table.
bd 0 Disable breakpoint #0.

2. bt

```
[root@host /root]# cat /proc/partitions
Entering kdb on processor 0 due to Debug Exception @ 0xc01845e3
Read/Write breakpoint #1 at 0xc024ddf4
kdb> bt
  EBP         Caller              Function(args)
0xc74f5f44 0xc0146166    get_partition_list(0xc74d8000)
0xc74f5f8c 0xc01463f3    get_root_array(0xc74d8000, 0x13, 0xc74f5f88,
0xf3, 0xc00)
0xc74f5fbc 0xc0126138    array_read(0xc76cd80, 0x804aef8, 0xc00, 0xc76cdf94)
0xbffffcd4 0xc0108b30    sys_read(0x3, 0x804aef8, 0x1000, 0x1000, 0x804aef8)
kdb> bp
Instruction Breakpoint #0 at 0xc0111ab8 (schedule) in dr0 is disabled on cpu 0
Data Access Breakpoint #1 at 0xc024ddf4 (gendisk_head) in dr1 is enabled on cpu
0 for 4 bytes
kdb> go
[root@host /root]#
```

3. ss

```
kdb> bp gendisk_head datar 4
Data Access Breakpoint #0 at 0xc024ddf4 (gendisk_head) in dr0 is enabled on cpu 0
for 4 bytes
kdb> go
[root@host /root]# cat /proc/partitions
Entering kdb on processor 0 due to Debug Exception @ 0xc01845e3
Read/Write breakpoint #0 at 0xc024ddf4
[0]kdb> ssb
sd_finish+0x7b:    movzbl 0xc02565d4,%edx
sd_finish+0x82:    leal      0xf(%edx),%eax
sd_finish+0x85:    sarl      $0x4,%eax
sd_finish+0x88:    movl      0xc0256654,%ecx
sd_finish+0x8e:    leal      (%eax,%eax,4),%edx
sd_finish+0x91:    leal      (%eax,%edx,2),%edx
sd_finish+0x94:    movl      0xc0251108,%eax
sd_finish+0x99:    movl      %eax,0xffffffc(%ecx,%edx,4)
sd_finish+0x9d:    movl      %ecx,0xc0251108
sd_finish+0xa3:    xorl      %ebx,%ebx
sd_finish+0xa5:    cmpb      $0x0,0xc02565d4
[0]kdb> go
[root@host /root]#

[0]kdb> ss
sys_read:      pushl    %ebp
SS trap at 0xc01274c1
sys_read+0x1:      movl      %esp,%ebp
[0]kdb> ss
sys_read+0x1:      movl      %esp,%ebp
SS trap at 0xc01274c3
sys_read+0x3:      subl      $0xc,%esp
[0]kdb> ss
sys_read+0x3:      subl      $0xc,%esp
SS trap at 0xc01274c6
sys_read+0x6:      pushl    %edi
[0]kdb>
```

五、Quagga 套件

Quagga 爲一路由軟體套件，此套件爲一共享軟體，可以在網路上自由下載使用。Quagga 由著名的路由軟體 Zebra 改版而來，支援 OSPFv2, OSPFv3, RIP v1 and v2, RIPng 以及 BGP-4 協定。

Quagga 在安裝完後，會在本機使用第 2601 號埠，提供使用者遠端登入設定。登入後爲 Quagga 之 view mode，只提供基本觀察指令，不能修改其設定。另一模式爲 configure mode，可供遠端設定更多 Quagga 設定。

Quagga view mode 指令列表如下：

Command	Description
echo	Echo a message back to the vty
enable	Turn on privileged mode command
exit	Exit current mode and down to previous mode
help	Description of the interactive help system
list	Print command list
quit	Exit current mode and down to previous mode
show	Show running system information
terminal	Set terminal line parameters
who	Display who is on vty

Quagga configure mode 指令列表如下：

Command	Description
access-list	Add an access list entry
banner	Set banner string
debug	Debugging functions (see also 'undebug')
enable	Modify enable password parameters
end	End current mode and change to enable mode.
exit	Exit current mode and down to previous mode
help	Description of the interactive help system
hostname	Set system's network name
interface	Select an interface to configure
ip	IP information
ipv6	IPv6 information
line	Configure a terminal line
list	Print command list
log	Logging control
no	Negate a command or set its defaults

password	Assign the terminal connection password
quit	Exit current mode and down to previous mode
router-id	Manually set the router-id
service	Set up miscellaneous service
show	Show running system information
table	Configure target kernel routing table
write	Write running configuration to memory, network, or ..
smux	SNMP MUX protocol settings

Ⅳ. 實驗方法

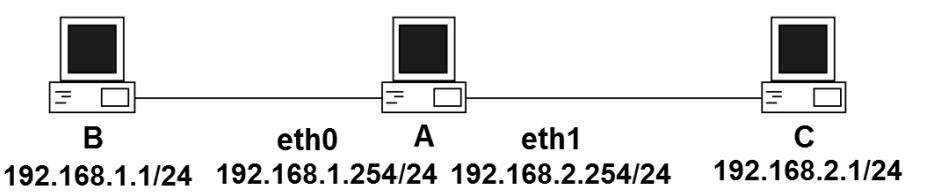
我們的實驗的過程大致可分爲三個部分，這三個部份並沒有一定的先後關係。KDB 爲一個 Linux 核心除錯工具，我們將利用它來了解核心對於網路封包處理流程。再使用 route 指令來對 Linux router 作設定，並安裝 Quagga 套件，讓 router 支援動態路由。最後再用簡單的 ping 指令來對實驗所建構的 Linux 路由器進行測試，確認該路由器的封包傳送功能是否正常運作。

Ⅴ. 實驗步驟

1. 安裝 Linux

   安裝 Linux 非常簡單，只需要用光碟或 FTP 等就可以灌好。坊間有許多參考書籍介紹安裝這部分，網路上也有很多網站有說明文件，在此就不贅述。

2. 設定並測試 Linux router
   1. 設定 Linux router 的二張網路卡的 IP 位址分別爲 192.168.1.254 和 192.168.2.254，netmask 皆爲 255.255.255.0
   2. 分別於兩張網卡之 interface 接上一 PC，IP 位址分別爲 192.168.1.1，GATEWAY 爲 192.168.1.254 和 192.168.2.1，GATEWAY 爲 192.168.2.254 netmask 皆爲 255.255.255.0
   3. 分別於 PC A 及 PC B，使用 PING 指令，測試 router 是否正常運作。
      於 ip 爲 192.168.1.1 之 PC 上，鍵入 ping 192.168.1.2，測試該主機是否有回應。反之於 192.168.1.2 之 PC 上，作相同測試。



圖一：Linux路由器：建構、測試與追蹤 實驗平台

3. 安裝核心與 KDB

依前面「Ⅱ.實驗設備」中所提及的方法取得 Linux 核心和 KDB。在/usr/src 下解開核心的原始碼，接著將 kdb-v4.4-2.6.16-common-5.bz2 以及 kdb-v4.4-2.6.16-i386-3.bz2 放到/usr/src/linux-2.6.16 這個新解開的目錄，bzip2 –d kdb-v4.4-2.6.16-common-5.bz2 、bzip2 -dkdb-v4.4-2.6.16-i386-3.bz2 解開，要 patch 之前最好先檢視一下 patch 檔，選定以 patch –p1 < kdb-v4.4-2.6.16-common-5 和 patch –p1 < kdb-v4.4-2.6.16-i386-3 指令進行 patch，最後便要進行核心編繹的動作。

編譯新核心的部份可以參照實驗二：Linux 下網路驅動程式追蹤的實驗步驟 2.，但在設定核心時有幾點要特別注意：1.確認所選用的網路卡裝置應能在核心版本(Linux-2.6.16)正確驅動。 2.需選用 CONFIG_KDB 、CONFIG_KDB_FRAMEPTR 和 CONFIG_KDB_OFF，才能開啓 KDB 的功能。

4. 使用 KDB 追蹤核心流程

安裝完核心並重新開機後，使用 echo “1” > /proc/sys/kernel/kdb 手動開啓 KDB module 之功能，任意時候都可以按 Pause 鍵來啓動 KDB。在啓動 KDB 後，我們設定 ip_rt_ioctl 這個函式中斷點並且在 shell prompt 下鍵入指令 route del default，這個指令有用到 kernel 中的 ip_rt_ioctl 函式，所以此時自動進入 KDB 中，我們可以透過 bt，ss 和 ssb 等指令觀察 kernel 的運作並將 ip_rt_ioctl 所呼叫過的函式紀錄起來【記錄 1】，另外在【記錄 2】中，觀察 ip_route_input。你也可以和/usr/src/linux 的 kernel source tree 交互比對驗證。

5.安裝和設定 Quagga 套件

1.請至 Quagga 官方網站[2]下載此套件 quagga-0.99.4-1.fc5.i386.rpm，使用 rpm 方式安裝套件

```
[root@localhost ]# rpm -ivh quagga-0.99.4-1.fc5.i386.rpm
Preparing...                ########################################### [100%]
   1:quagga                 ########################################### [100%]
```

2.設定 Zebra 並且啓動 Zebra

```
[root@ localhost ]# vi /etc/quagga/zebra.conf
hostname linux.router    ←設定此路由器之主機名稱
password nctu            ←設定密碼爲 nctu
enable password nctu     ←啓動密碼
log file zebra.log       ←將所有 zebra 產生的資訊存到 zebra.log 中
[root@ localhost ]# /etc/init.d/zebra start      ←啓動 zebra
[root@ localhost ]# netstat –tunlp               ←查詢 zebra 是否正確啓動
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address      Foreign Address      State      PID/Program name
tcp        0      0 127.0.0.1:2601    0.0.0.0:*            LISTEN     6422/zebra
```

3.登入 Quagga 並秀出目的路由資訊

```
[root@localhost ~]# telnet localhost 2601
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.

Hello, this is Quagga (version 0.99.4).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

User Access Verification

Password:
localhost> show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.2.0/24 is directly connected, eth0
C>* 192.168.2.0/24 is directly connected, eth1
K>* 169.254.0.0/16 is directly connected, eth1
```

4.在 Quagga 中加入靜態路由

```
esslab16.cis.nctu.edu.tw> enable        ←進入 enable mode
Password:nctu                           ←輸入密碼 nctu
esslab16.cis.nctu.edu.tw# configure terminal      ←進入 configure mode
新增一靜態路由
esslab16.cis.nctu.edu.tw(config)# ip route 192.168.100.0/24 eth0
esslab16.cis.nctu.edu.tw(config)# exit            ←離開 configure mode
esslab16.cis.nctu.edu.tw# show ip route           ←秀出路由資訊
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.1.0/24 is directly connected, eth0
C>* 192.168.2.0/24 is directly connected, eth1
S>* 192.168.100.0/24 [1/0] is directly connected, eth0 ←靜態路由資訊加入成功
```

```
K>* 169.254.0.0/16 is directly connected, eth1
```

5.設定 ripd 服務

```
[root@localhost]# vi /etc/quagga/ripd.conf
hostname linux.router      ←設定 Router 的主機名稱
password nctu              ←設定密碼爲 nctu
router rip                 ←啓動 Router 的 rip 功能
network 192.168.1.0/24     ←指定監聽此網域
network eth0               ←指定監聽此介面
network 192.168.2.0/24     ←指定監聽此網域
network eth1               ←指定監聽此介面
version 2                  ←啓動 RIPv2 服務
log stdout                 ←在螢幕輸出標準輸出的資料

[root@localhost]# /etc/init.d/ripd start

[root@localhost]# netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address    Foreign Address State    PID/Program name
tcp        0      0 0.0.0.0:2602     0.0.0.0:*        LISTEN 21373/ripd
```

VI. 實驗記錄

記錄	內容
1	觀察 ip_rt_ioctl 所呼叫的函式。
2	觀察 ip_route_input 所呼叫的函式。

3	如何用 route 指令，刪除 192.168.1.0 端之 pc 對 192.168.2.0 之路由?並使用 ping 指令測試對 routing table 之修改是否成功。
4	於 192.168.1.1 架設 ftp server，並使用 PC:192.168.1.2 爲 client 測試傳輸速度，並記錄最高傳輸速度。
5	步驟 2-3 中，使用 ping 指令測試 router 功能是否正常，請將 ping 之結果畫面截取下來，並紀錄之。
6	實驗步驟中登入 quagga，並 show 出當時 router 之路由資訊，請紀錄實驗之 router 的完整路由資訊。
7	使用 netstat – tulnp 指令查詢主機之有開啓服務的埠號，並紀錄下來。

Ⅶ. 問題與討論

一、請問如果需要製作具有 IP Masquerade 和 Firewall 功能的 Linux 路由器，有那些地方是需要修改或新增的？而那些地方又是應該要注意的？

二、如果希望爲這台 Linux 路由器加上具有 QoS 的功能，請問有那些地方需要增加或修改，又那些地方具有這些資訊？（可以列出 paper 或者網站）

三、請問在 Linux kernel 中，每個封包進入之後，路由查詢的處理流程爲何，由那幾個函式處理?

四、在 Linux kernel 中，除了一般的路由表以外，還有快取路由表。試比較這兩個路由表之結構。

五、在安裝 KDB 這個套件的步驟中，版本的相依性絕對地重要，試從你實驗過程中的觀察，說明爲何版本相依性如此重要。

六、自問自答。(可以是您在操作所遇到的問題並解決的方法，或是新的啓示和想法)

VIII. 參考文獻

[1]. Fedora Project, sponsored by Red Hat,http://fedora.redhat.com/.

[2]. The Linux Kernel Archives,http://www.kernel.org/.

[3]. SGI - Developer Central Open Source | KDB, http://oss.sgi.com/projects/kdb/.

[4]. Quagga Software Routing Suite, http://www.quagga.net/.

[5]. Alessandro Rubini, "Linux Device Drivers", O'Reilly & Associations, Feb 1998.