網上窺探—小工具大用途

潘貞如、林盈達

國立交通大學資訊科學系

前言

在五花八門的網路世界裡，有各式各樣的指令或工具來供應每個人不同的需求，你瞭解這些網路工具內部如何運作嗎？當你上網認識許多新朋友，或許會想留意朋友的最新動態，此時你會利用工具查詢嗎？你可曾知道 ping, finger, traceroute 的功用？在這篇文章中，我們將介紹這些網路上簡單的小工具，使用 ping 來查詢某一站台是否存在、finger 可得知使用者資料及利用 Traceroute 來瞭解到某一目地位址的路徑狀況。這三個窺探機器、個人、路徑的工具究竟如何運作是我們介紹的重點。

1.Ping — 窺探機器

一般人最常使用的網路應用包括 telnet、WWW 與 mail 等，而當你在利用這些軟體時，是否曾發生過想連上某個站台卻一直連不上的情況？是站台掛了？還是目前系統因負荷太大、已達飽和狀態而暫時停止新使用者的加入？這個時候，ping 就派上用場了！PING 全名是 Packet InterNet Groper [1]，此指令主要用途為查詢站台是否存在，也就是說看看站台是否有連接於網路上並工作中。若此站與網路相通，則 ping 會回傳表示站台存在的訊息；相反地，當你與某一站之間的網路並不相連時，下 ping 這指令將會出現此站沒有回應的訊息，表示這兩點之間的網路是中斷的，或者此站台掛掉了！因此，經由 ping 這個小程式你便可以瞭解目前的狀況，如果站台存在，那你只好多試一下，等網站的使用量小一些再進去。

下面即為兩個 ping 的範例。範例一是查詢 140.117.5.15 這個位址，執行

ping 140.117.5.15 後，我們發現執行結果是 Request timed out.。這個結果顯示所查詢的位址並沒有回應，即表示它目前並未與網路連接，可能是掛掉，或者此位址目前根本無人使用。相同地，範例二亦為查詢一個位址：140.113.88.254。由範例中可看到其執行結果並未出現 Request timed out，相反地卻出現由 140.113.88.254 所回應的訊息。這個訊息表示你所發出去的封包被 140.113.88.254 接收，並依照封包的協定，回應訊息給發出封包的位址，這在稍後我們將會介紹。

```
C:\>ping 140.117.5.15
Pinging 140.117.5.15 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.
```

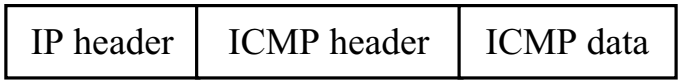
範例一、對方無回應的情況

```
C:\>ping 140.113.88.254
Pinging 140.113.88.254 with 32 bytes of data:

Reply from 140.113.88.254: bytes=32 time=1ms TTL=255
Reply from 140.113.88.254: bytes=32 time=2ms TTL=255
Reply from 140.113.88.254: bytes=32 time=1ms TTL=255
Reply from 140.113.88.254: bytes=32 time=1ms TTL=255
```

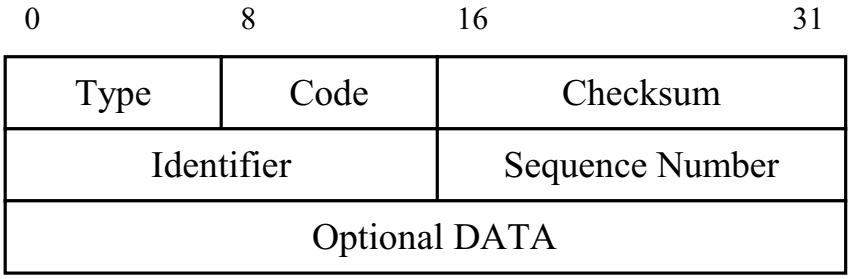
範例二、對方存在且有訊息回應的情況

知道了 ping 大略的用途後，接著來看 ping 這個程式是如何運作的。就像傳遞一段聲波，當碰撞到物體時聲波會自動返回，ping 即是利用此原理。程式主要包含兩個部分，利用 ICMP (Internet Control Message Protocol) 其中的兩個訊息格式（Echo Request 與 Echo Reply 這兩個訊息）分別作傳送及接收的工作。有關 ICMP 訊息的規格如圖一及圖二[2][3]。 ICMP 訊息是包含在 IP 的封包裡傳送，如圖一。ICMP 協定軟體是每一個 IP 節點都要支援的，因

此每個 IP 節點都能傳送及處理 ICMP 的訊息。ICMP 訊息最終目的地並不是目的機器的應用程式或使用者，而是目的機器上的 IP 軟體，也就是說，當我們收到一個 ICMP 訊息時，目的機器上的 ICMP 軟體會自動去處理回應。因此 ping 利了用 ICMP 協定的這些特性，而沒有依賴 TCP 和 UDP 來作為傳送的協定。圖三即表示這些網路協定間的階層關係。[4]



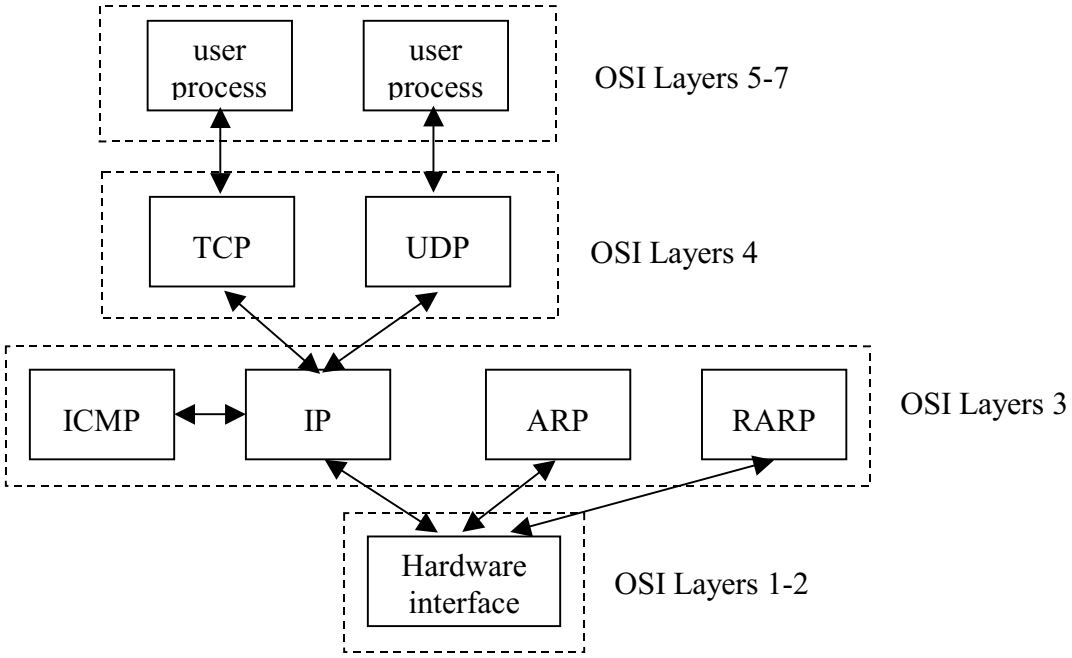
圖一、ICMP 訊息



Type : 8 for echo request message
0 for echo reply message

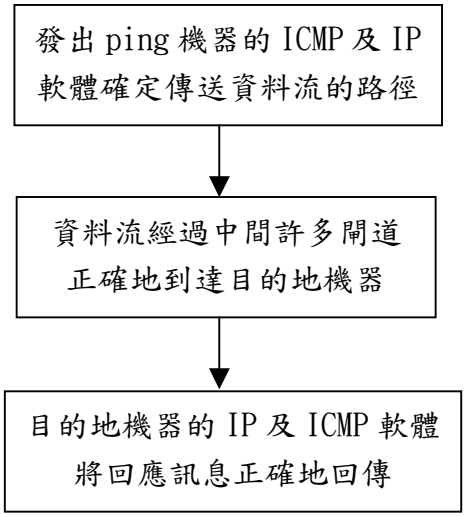
圖二、ICMP echo request 或 reply 之訊息格式

圖二中可看出，echo request 與 echo reply 的型態是由 Type 來決定。echo reply 這個訊息的資料部分通常是 echo request 傳過來時的內容拷貝。通常我們會將傳輸時的時間寫在 DATA 上，以方便計算封包回傳後這之間所花的時間。而 Identifier 與 Sequence number 是可用來幫助傳送者辨認相對應的 reply 訊息。Optional Data 為一個長度可任意變動的區域，此區域包含欲回傳給原來傳送者的一些資料，例如剛剛所說封包傳送中的傳輸時間。

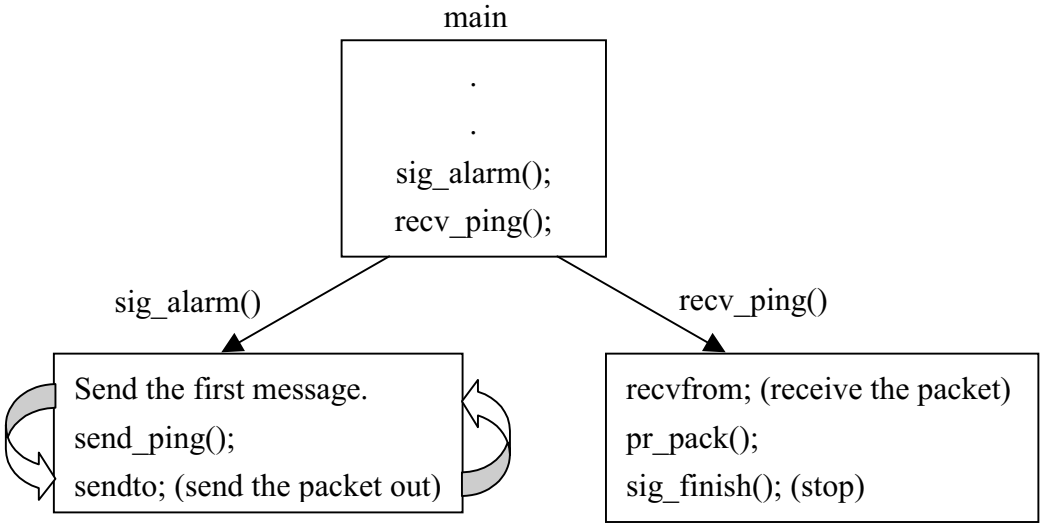


圖三、網路協定的階層關係

由於 echo request 與 echo reply 這兩個 message 是隨著 IP 資料流遊走，因此若一個點能收到回應的訊息即表示這兩點之間傳輸的管道和所有提供的協定是暢通的。其間的流程如圖四：



圖四、Ping 的流程



圖五、ping 程式架構

程式主要架構是經由呼叫兩個副程式來運作，即剛剛所說的傳送與接收，sig_alarm()、recv_ping()。主程式抓到想要 ping 的位址後，便呼叫 sig_alarm()開始傳送訊息。 sig_alarm()被主程式呼叫後，此時這個 sig_alarm()便為一個 signal handler 而依情況以遞迴方式呼叫自己，而呼叫時間是依所需時間自訂，如每秒呼叫一次。sig_alarm()傳送 message 前需將 ICMP 格式寫好，這一部份則由 send_ping()負責。send_ping()在 sig_alarm()這程式前端將準備工作做好，隨即可用 sendto 這個函式將封包由 UDP 層送出去。

在接收回傳訊息的部分是由 recv_ping()負責。這是一個無窮迴圈，從 socket 端用 recvfrom 這個函式隨時將封包接收進來。當我們收到封包時，立即呼叫 pr_pack()來處理並判斷是否為 ICMP 封包。pr_pack()會檢查每一個進來的封包，檢查這些封包是否由自己送出去訊息所回傳的。當我們做完所有的事，即由 sig_finish()來收尾，印出統計的資料，程式也結束。

2.Finger — 窺探個人

當你上網時，除了查詢資料外，亦能與三五好友聊聊天、交換資訊，但使用者上線時間無法預知，當雙方不能巧遇時，人總是有很大的好奇心，會想看看某人上次上站的時間究竟是何時，或由其計畫檔得知目前的狀況，而這個時候就會用到 finger 了。也或許會發生這樣的情形，當某一使用者在網路上發表激進的言論而引起你的注意時，同樣地，用 finger 便可將大概地底細摸個清楚了。下面是一個 finger 的範例。

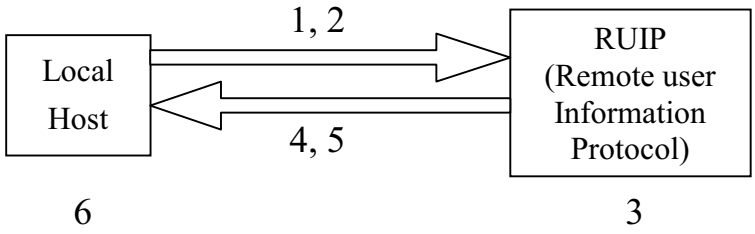
```
cissol3:[~]$ finger is85028

wheel (is85028)
Home: /home6/is85/is85028
Shell: /bin/tcsh
No unread mail.
wheel (is85028) is not presently logged in.
Last seen at cissol4 on Thu Oct 29 00:58:55 1998

No plan.
cissol3:[~]$
cissol3:[~]$ finger tslc@cs.tku.edu.tw
[cs.tku.edu.tw]
Login name: tslc
Directory: /home/faculty/tslc           Shell: /bin/csh
Last login Fri Dec  4 18:47 on pts/6 from pc80.dorm2.tku.e
New mail received Tue Dec 15 19:51:46 1998;
  unread since Fri Dec 11 18:06:21 1998
No Plan.

cissol3:[~]$
```

範例三、finger 的執行結果

範例三是在工作站中分別執行兩次 finger，在 finger 後輸入你所想查詢的 id，便可得到此 id 的相關資料，如他所登錄的名字、計劃檔、何時上線等資訊。

Finger 也是一個簡單的工具程式，提供了與遠端使用者資訊的一個介

面。當你想查詢時，finger 會依照你所輸入的參數，提供與參數相關的使用者資料，例如，使用者的名字、綽號、計畫、上站時間、或上站地點、終端機型態等。Finger 是跑在 TCP 協定之上，使用 TCP 第 79 埠來建立彼此的連線。



1. 由 Finger 埠建立一個新的連線(port 79)
2. 送 finger 的要求給遠端使用者資訊協
3. RUIP 處理這個 finger 的要求
4. RUIP 回傳資訊給 local host
5. 通知準備關閉連線
6. local host 收到 RUIP 所有的資訊，並關掉連線

圖六、Finger 流程

圖六即為整個 finger 的程式流程，主機發出 finger 要求時，程式會先建立一段與遠端主機間的一條連線，此連線是利用遠端伺服器的第 79 埠來運作。建立連線後，主機立即將 finger 的要求傳送過去給遠端使用者資訊協定。此協定便開始依照傳送過來的參數要求，查詢相關的使用者資料。當 RUIP（Remote User Information Protocol）處理完這些使用者資訊，當然就必須回傳給發出要求的主機了，而這個時候，也必須準備關閉彼此間的連線。此時，主機收到資料後，便將兩端的溝通管道關閉[5][6]。

3.Traceroute — 窺探路徑

這也是一個簡單的查詢指令。有時候，我們可能會發現與某個站台之間的連接似乎有困難存在，如 mail 到另一地會發生問題，這個時候我們應該找出這段路徑間究竟是哪裡出錯，找出問題點才能修復它，而這就可以靠 Traceroute 來做了！若你想查詢從你的位址出發到某一站之間究竟經過多少

的閘道，其路徑及延遲究竟如何，Traceroute 可幫你一一查出。範例四、範例五各為由交大到英國與美國某一點 traceroute 的結果。

```
cissol3:[~]$ traceroute www.cam.ac.uk
traceroute to www.cam.ac.uk (131.111.8.46), 30 hops max, 40 byte packets
 1  23-router (140.113.23.254)  2 ms  2 ms  1 ms
 2  140.113.60.11 (140.113.60.11)  2 ms  2 ms  3 ms
 3  CC7513a-atm804.nctu.edu.tw (140.113.247.253)  2 ms  3 ms  2 ms
 4  203.72.38.101 (203.72.38.101)  12 ms  6 ms  4 ms
 5  210.70.55.2 (210.70.55.2)  221 ms  220 ms  220 ms
 6  * 12.127.196.201 (12.127.196.201)  217 ms  215 ms
 7  * gr1-a350s5.sffca.ip.att.net (192.205.35.21)  215 ms  217 ms
 8  nr1-h02.napsf.ip.att.net (192.205.31.42)  213 ms *  210 ms
 9  * * *
10  207.45.223.57 (207.45.223.57)  229 ms  251 ms *
11  * * *
12  207.45.223.53 (207.45.223.53)  284 ms * *
13  207.45.223.113 (207.45.223.113)  314 ms  305 ms *
14  207.45.209.250 (207.45.209.250)  319 ms  323 ms  322 ms
15  * * *
16  207.45.223.41 (207.45.223.41)  329 ms * *
17  cust-gw.Teleglobe.net (207.45.215.166)  328 ms * *
18  * * 193.62.157.5 (193.62.157.5)  408 ms
19  * external-gw.ja.net (193.63.94.40)  418 ms  411 ms
20  * london-core.ja.net (146.97.251.58)  411 ms *
21  * * *
22  * route-cent-0.cam.ac.uk (131.111.1.1)  414 ms  411 ms
23  * route-cent-8.cam.ac.uk (131.111.2.8)  441 ms *
24  * * *
25  cygnus.csx.cam.ac.uk (131.111.8.45)  425 ms  412 ms *
```

範例四、由交大到英國 www.cam.ac.uk Traceroute 結果

```
cissol3:[~]$ traceroute 128.122.20.15
traceroute to 128.122.20.15 (128.122.20.15), 30 hops max, 40 byte packets
1  23-router (140.113.23.254)  2 ms  1 ms  1 ms
2  140.113.60.11 (140.113.60.11)  2 ms  1 ms  2 ms
3  CC7513a-atm804.nctu.edu.tw (140.113.247.253)  2 ms  2 ms  2 ms
4  203.72.38.101 (203.72.38.101)  6 ms  4 ms  8 ms
5  210.70.55.2 (210.70.55.2)  214 ms  235 ms  250 ms
6  12.127.196.201 (12.127.196.201)  237 ms * *
7  gr1-a3100s5.sffca.ip.att.net (192.205.35.17)  223 ms  221 ms *
8  sl-gw14-stk-1-0-0-T3.sprintlink.net (144.228.192.1)  225 ms  231 ms  216 ms
9  * sl-bb23-stk-4-2.sprintlink.net (144.232.4.65)  220 ms *
10  * sl-bb10-pen-6-0.sprintlink.net (144.232.8.177)  288 ms *
11  * at-bb1-pen-6-0-0.appliedtheory.net (144.232.5.114)  351 ms  297 ms
12  * at-bb3-nyc-0-1-0.appliedtheory.net (169.130.1.126)  290 ms *
13  * at-gw2-nyc-0-0.appliedtheory.net (169.130.10.3)  354 ms *
14  * * *
15  * NYUGWA-FA1-0-0.NYU.NET (192.76.177.65)  680 ms *
16  * WWHGWC-FDDI-1-0.NYU.NET (128.122.253.74)  655 ms  598 ms
17  SPARKY.CS.NYU.EDU (128.122.20.15)  632 ms  350 ms *
```

範例五、由交大到美國 128.122.20.15 Traceroute 的結果

基本上，封包在傳遞過程中，每經過一個 hop，其中的有一個 TTL（Time to Live）欄位，會被依序遞減，原本是為了防止因過長的傳遞路徑，導致此封包失去時效，甚至無止盡的傳遞下去，而設定的一個變數。而 Traceroute 便利用了這個特性，藉由巧妙的設定傳送出去封包的 TTL 值，來達到收集沿路各 hop 的資訊。

Traceroute 首先傳送出一個 TTL 值為 1 的 UDP 封包，如此當第一個 hop 收到這個封包，而欲傳送下去時，便會發現其 TTL 值已然被遞減為零，換句話說，此封包已經過時，因此此 hop 便依照處理過時封包標準的程序，產生一個 ICMP 協定格式的”time exceeded”訊息回到發出此封包的 host，也就是執行 Traceroute 的 host。而 Traceroute 便利用傳回的這個封包所夾帶的資訊，如其中第一個 hop 送出 ICMP 訊息時，所設定的 source IP address,以及計算送出 UDP 封包及收到”time exceeded”訊息封包的時間差，來得知所在 host 與此傳送路徑之第一個 hop 間所存在的 delay time。接著再送出 TTL 為 2 的 UDP 封包，按照上述的過程，此第二個封包會在路徑中的第二個 hop 失去時效，也因此我們又可以得到第二個 hop 的相關資訊，如此依序進行下去，一直到抵達目的地為止，如此從這頭到那一端的路徑便可一目了然[7]。

有幾點是必須特別注意的，第一，因為我們不想讓目的主機真的去處理這個封包，因此將目的埠設為不合理的值。第二，正如 TTL 值的對於一般 IP 封包的作用般，我們也必須防止 Traceroute 為了找出目的地，無限制的探詢下去，因此依序累增的 TTL 值也必須有個上限。

圖六是 Traceroute 的演算法。先利用一個旗標 Flag_stop 來判定目的地是否已經到達(1)。我們可自行設定最大的 TTL 數，而這個程式會重複執行一直到到達目的地，或 TTL 值已達到所設定的最大值時，程式停止(2)。首先，程式先將目前的時間記錄到 T1(3)。TTL 初始值設為 1，並將封包的欄位填寫完整後送出封包(4)，並等待 hop 的回傳訊息(5)。收到正確的回傳封包時，我們再記錄其到達的時間以做為統計(6)。第 7 個步驟則是確定是否已經傳到了目的地，若達目的地，則程式將會結束。接下來的步驟即處理沿路上的回傳封包，所收集到的統計資料，並將結果顯示出來(8)(9)。

```
1.  Flag_stop=0
2.  While ( I < maxTTL )AND(flag_stop==0)
    3.  get current time -> T1
    4.  Send the packet of   TTL = I
    5.  Wait for reply
    6.  Get current time -> T2
    7.  if (Rx pkt's source addr == Tx pkt's target addr. )
        then flag_stop=1
    8.  Compute the difference between T1 and T2
    9.  Display the relate information about the path
10. End of while
```

圖六、Traceroute 演算法

■ 4.比較與結論

認識了 ping，finger，traceroute 這三個網路工具程式之架構後，我們對這三個程式依其 Protocol、應用層面、及其共同與不同的地方做一些比較，如表一所示。在 Protocol 方面，ping 是利用 ICMP 協定而 traceroute 需利用 UDP 及 ICMP 協定來傳送封包，而 finger 則必須到遠端的 RUIP 協定去做查詢，因此 finger 是架構在 TCP 上。在這三種工具程式中，ping 與 traceroute 的共同點較多，他們皆為傳送一段特定協定的封包，由網路中的軟體自動回應訊息。與 ping、traceroute 較不相同的 finger 則是讀取輸入的參數後，將這些參數傳送到遠端的使用者伺服程序去取得資訊，因此必須與遠端建立連線。

比較	**Ping**	**Finger**	**Traceroute**
Protocol	利用 ICMP 協定 無須使用其他協定	架構於 TCP 上 需使用 RUIP 協定	利用 UDP 及 ICMP 無須使用其他協定
應用	查詢主機是否能連通	查詢使用者資料	查詢與遠端主機

			之間的路徑
共同性	1. Ping 與 Traceroute 不需要遠端的伺服程序 2. Ping 與 Traceroute 皆是傳送一段包含 ICMP 協定的封包，並等待回傳封包值。 3. Ping 與 Traceroute 不需建立兩端之間的溝通管道		
不同	Finger：被查詢的遠端有伺服程序(finger daemon)以供發出要求的主機查詢 Finger 需先建立起兩端之間的連結管道。		

表一、綜合比較 ping、finger、traceroute

大致上介紹完上面一些簡單的小技巧，不知道大家對網路工具程式是否有大概的瞭解。網路是目前最便利的一項資訊瀏覽及交流工具，在今天網路多元化的時代，網路應用無奇不有，新奇的事物也因應而生，相信還有許多有趣的東西等著我們去開創。而我們今天若能熟悉一些基本的小技巧，不但對網路的運作方式有更進一步的了解，善加利用這些小技巧，相信更能掌握想要知道的資訊，甚至可自行發展一些小程式作為應用或工具，更能自由地遨遊在網路的世界中。

參考資料

[1] W. Richard Stevens, “UNIX Network Programming”, Prentice Hall, 1991

[2] W. Richard Stevens, “Advanced Programming in the UNIX Environment”, Addison Wesley, 1992

[3] J. Postel, “Internet Control Message Protocol”, RFC 792, Sep 1981

[4] Douglas E. Comer, “Internetworking with TCP/IP Volume Ⅰ: Principles,

Protocols, and Architecture", Second Edition, Prentice Hall, 1991

[5] K. Harrenstien, "NAME/FINGER", RFC 742, Dec 1977

[6] D. Zimmerman, "The Finger User Information Protocol", RFC 1288, Dec 1991

[7] G. Malkin, "Traceroute Using an IP option", RFC 1393, Jan 1993