Chapter 9 – IP Routing

Two principal functions of the network layer

routing
congestion control

A routing hierarchy exists in IP

Routers connecting autonomous systems – Exterior gateway protocols
Routers within Autonomous system – Interior gateway protocols
Host level routing

Routing approaches

Class based (Host level and IGPs)
    Class A, B, C networks
    Subnets and subnet masks are significant
Classless (EGPS and IGPs)
    Prefix/Prefix len

Host level routing

Driven by next hop lookup table
Destination column is the lookup key
Gateway identifies the next hop
Interface specifies device to which packet should be queued.
But the gateway address is still significant .. note multiple gateways on *le0*

A sample routing table

```
/local/cdrom/inet/rfc ==> netstat -nr
Routing tables
Destination      Gateway          Flags    Refcnt Use    Interface
130.127.70.9     130.127.48.2     UGHD     0      8          le0
127.0.0.1        127.0.0.1        UH       1      2470       lo0
130.127.70.2     130.127.48.2     UGHD     0      2127       le0
130.127.66.12    130.127.48.4     UGHD     0      57         le0
130.127.70.12    130.127.48.2     UGHD     0      2          le0
130.127.66.15    130.127.48.4     UGHD     0      1247       le0
default          130.127.48.1     UG       0      83         le0
130.127.48.0     130.127.48.24    U        28     297732     le0
```

Routing table elements

Destination – A host, network, or default entry
Gateway – The IP Address of the next hop
Flags
- U – Route is up
- G – Route is to a router (not direct to destination).
- H – Route is to a specific host (as opposed to a network)
- D – Route established by an ICMP redirect
- M – Route was modifed by a redirect

Refcnt – # of connections (TCP) to destinations using this route.
Use – # of packets routed.
Format will vary from OS to OS

Distinguishing between G and H

- H means a "host" system *anywhere* on the internet
- ~G means this route is direct to the target host ... doesn't pass through any gateway
- G means the route does pass through a gateway.. Thus GH are *not* mutually exclusive

Table lookup priority order (host level)

1 – Matching host
2 – Matching network
3 – Default route

Sources of routing table entries

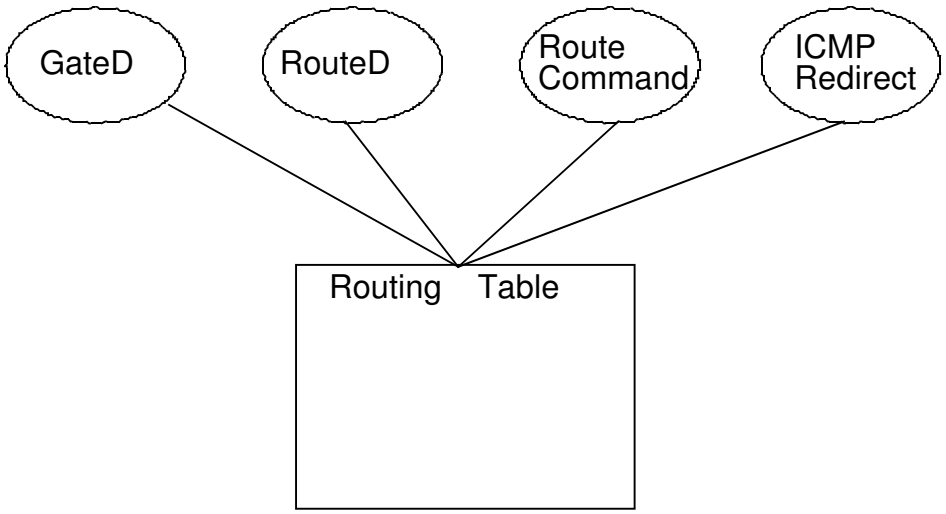
Route command will be used in boot scripts to construct

localhost
localnet
default entries

```
route add -host 130.127.48.99 gw 130.127.48.24 dev eth0
```

Redirects may be received when there are multiple routers on the local net

Hosts may run Gated or RouteD to exchange routes with local routers (but generally don't)



Minimal routing table requirements:

Isolated host
- Only the loopback address

Isolated LAN
- Loopback + network address

LAN Connected to Internet
- Loopback + network (the LAN itself) + default (the gateway)

Example in book shows that it may be possible to "reverse engineer" the subnet mask from a routing table entry.

```
130.127.48.248   130.127.48.250  U         28     297732     le0
```

Address is a network not a host (no H flag)
Trailing zero bits are turned off in the network mask
Network address in hex is 827F30F8 and mask is FFFFFFF8
130.127.48.254 (827F30FE) matches mod netmask and is routeable
130.127.48.240 (827F30F0) is not.

Example in the book on "N" ways to ping (or ftp yourself)

ping jmw3
ping 130.127.48.118
Sent to ethernet driver which forwards it to loopback driver
Why??
ping localhost
ping 127.0.0.1
Sent direct to loopback driver.

Default routes are used not only in hosts but also in Interior Gateways
This can be demonstrated by pinging a non existent address

```
ping 192.82.148.1
ICMP Host Unreachable from gateway 192.221.42.100
    for icmp from jmw (130.127.48.24) to 192.82.148.1
no answer from 192.82.148.1
```

The ping received the unreachable notification from `192.221.42.100`

We can see how many routers forwarded the packet using a default route by running a tracroute to the router that finally dropped it.

```
traceroute to 192.221.42.100 (192.221.42.100), 30 hops max, 38
 0  citron.cs.clemson.edu (130.127.48.1)  0 ms  0 ms  0 ms
 1  citron.cs.clemson.edu (130.127.48.1)  0 ms  0 ms  0 ms
 2  130.127.44.1 (130.127.44.1)  0 ms  0 ms  0 ms
 3  130.127.2.1 (130.127.2.1)  0 ms  31 ms  0 ms
 4  clemson-gw.clemson.edu (130.127.8.5)  0 ms  0 ms  0 ms
 5  gnu1-clem-c1.sura.net (192.221.4.33)  31 ms  31 ms  0 ms
 6  atu1-gnu1-c3mb.sura.net (192.221.1.1)  31 ms  31 ms  63 ms
 7  cpe1-fddi1.Atlanta.mci.net (192.221.42.100)  62 ms * *
```

Host forwarding

Often a bad idea for host's
to participate in routing exchanges or
to forward IP datagrams

Host forwarding is *never* useful unless the host has at least two interfaces!

Cases where host forwarding can be useful

Host is a ppp server
Host is a NAT gateway
Host is a gateway to a private network

A simple PPP connection

From *jmw4* dial and log in to *jmw2* and su to root

On *jmw2* run the command

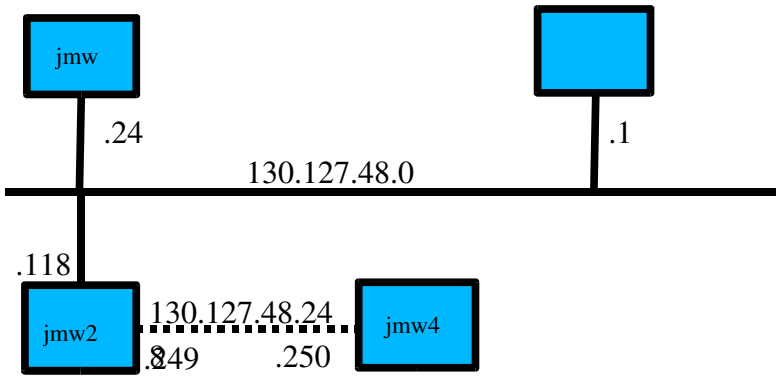
```
pppd noauth
```

Open another window on *jmw4* and run the commands

```
pppd /dev/cua0 38400 130.127.48.250:130.127.48.249  \
      netmask 255.255.255.248 debug noauth
```

When the pppd's connect, a routing table entry for *jmw2* is created in *jmw4*'s routing table.... but we still have some problems

```
Kernel IP routing table
Destination     Gateway    Genmask          Flags Metric Ref    Use Iface
jmw2            *          255.255.255.255 UH    0      0        0 ppp0
127.0.0.0       *          255.0.0.0       U     0      0        0 lo
```



Testing the connection

Then try pinging both directions

```
jmw2> ping 130.127.48.250
jmw4> ping 130.127.48.249
```

It works OK.

Test pinging host jmw on the LAN beyond the router jmw2

```
jmw4> ping 130.127.48.24
```

It doesn't work (why not?).
There is no entry for 130.127.48.24 in jmw4's routing table telling it to use jmw2

Testing the *jmw2's* ability to route *jmw4's* packets

Modify the *jmw4's* routing table to add a new host

```
jmw4> route add -host 130.127.48.24 gw 130.127.48.249
                                    dest address      router address
```

Now try ping again

```
jmw4> ping 130.127.48.24
```

No response is received...

The problem is diagnosed by running a packet logger on *jmw2* or *jmw*

The ping arrives at *jmw2*
It will or will not be forwarded depending on whether forwarding is enabled

```
echo "1" > /proc/sys/net/ipv4/ip_forward
```

When it reaches *jmw*, a ping reply will be scheduled to be sent to `130.127.48.250`
This address will match the local net entry in *jmw's* routing table
So *jmw* will *arp* for `130.127.48.250`
Since `130.127.48.250`is not on the ethernet there will be no arp reply
Thus the ping response will be dropped by the link driver

Resolving the reverse route problem

Possible Approach 1..

Add a route to *jmw*'s routing table

```
jmw> route add –host 130.127.48.250 gw 130.127.48.249
                                dest address        router address
```

Disadvantage

May not work... jmw2 may not respond to ARP requests for .249 on .118.
How about if we changed 249 to 118 (*jmw2*'s enet interface).. this should work.

Approach 2..

Use proxy arp

```
jmw2> arp –s 130.127.48.250 0:20:af:f:6f:3c  pub
                    Real IP address    Proxy E–net addr    Publish
                        (remote)          (jmw2)
```

Now ping by address to 130.127.48.24 will work.

Obtaining access to full IP routing capability and name services

Try a ping by name and ping of another host

```
jmw4> ping 130.127.48.22
jmw4> ping jmw
```

No response is received (either time). (Why not?)
No usable "default" route for

Unknown hosts
DNS queries (How does one find a name server?)

Solution

```
jmw4> route add default gw 130.127.48.249
```

Outbound from `jmw4`

The *entire* internet is available by name or address using any command.
telnet, ftp, etc.

Inbound to `jmw4`

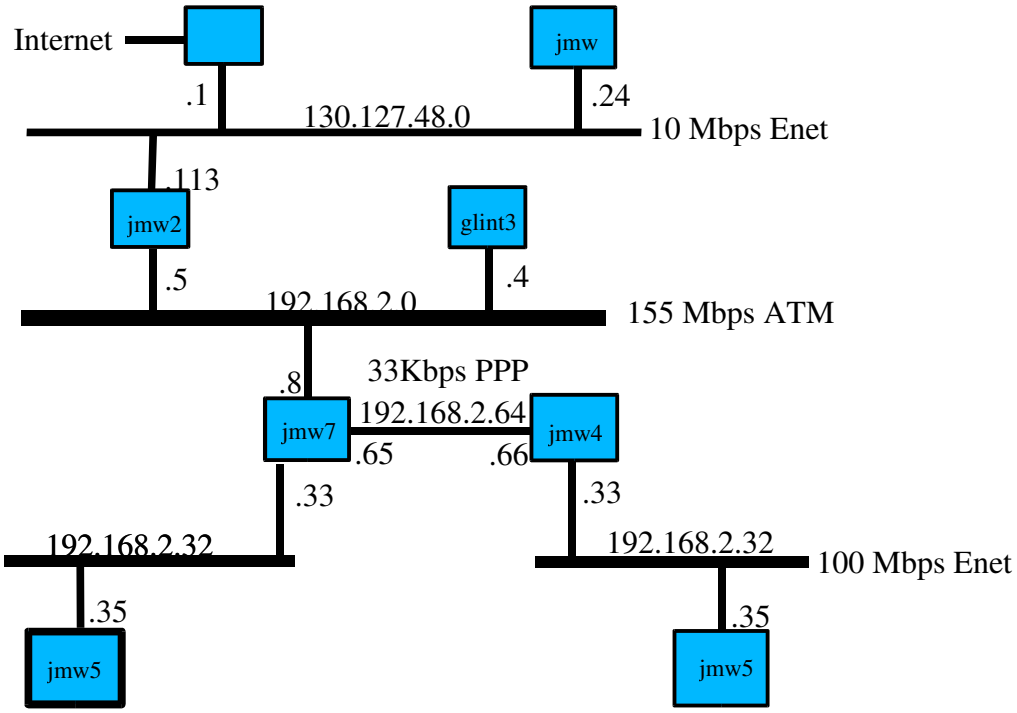
`jmw4` is available by address to the *entire* internet (why?.. no dns entry).
To the rest of the world it looks like its on net 130.127.48

Example of more complex host level routing

Net	130.127.48.0	is the departmental LAN
Net	192.168.2.0	is the 4th floor private ATM LAN
Net	192.168.2.64	is a PPP link between my home and office
Net(s)	192.168.2.32	are 100 Mbps Enets that host my notebook PC
Host	jmw2	is a network address translation (NAT) gateway

Objectives... without use of *any* routing daemons or protocols.. allow

glint3 and *jmw7* to contact *jmw*
jmw to contact *glint3* a*nd jmw7*
jmw4 to contact *glint3, jmw,* and the rest of the internet
jmw5 to contact *glint3, jmw,* and the rest of the internet
jmw and *glint3* to contact *jmw5*



Tools that will be employed in the solution

The route command
Proxy arp
Network address translation (NAT)
ATM LAN Emulation (LANE)

ATM LAN Emulation

Provides broadcast and ARP services that support IP over the ATM network

The network address translation gateway

translates source IP address of outbound packets from 192.168.2.x hosts to 130.127.48.113.
translates source port address to a new value which it remembers in NAT table

Source Proto	***Source Port***	***Source IP***	***New Source Port***	***Dest IP***	***Dest Port***
ICMP		192.168.2.7		155.2.3.4	
TCP	4155	192.168.2.2	1028	150.1.2.13	21
TCP	4155	192.168.2.3	1029	150.1.2.13	21
TCP	5523	192.168.2.3	1030	150.1.2.13	21
UDP	4444	192.168.2.4	1025	150.1.2.13	21
ICMP		192.168.2.6		155.2.3.4	

When an incoming packet arrives, the protocol id and destination port are used as a lookup key
Destination IP and Port are replaced using the values stored in the NAT table
Packet is then forwarded to the destination on the ATM LAN.

How do NAT gateways deal with ICMP packets (ping) ???

To enable NAT on jmw2

iptables

Hosts on the ATM LAN simply configure their default router to be the NAT gateway

```
route add default gw 192.168.2.5 dev lec0
```

After this is done all hosts on the ATM LAN can reach the entire internet.

Reaching the ATM LAN from *jmw*

One of the principal uses of the NAT gateway is as a firewall..
In that role it is intended to *prevent* access to the hosts behind it.
However, the following command *will* permit access from *jmw* to the ATM LAN

```
jmw> route add -net 192.168.2.0 130.127.48.113
```

How does the NAT gateway *avoid* translating return traffic from the ATM LAN???
Note that this "trick" works *only* for hosts on the 130.127.48.0 LAN

Configuring the hosts on the 100 Mbps LAN and ppp link

These must use *jmw7* as their default gateway's

```
jmw4> route add default gw 192.168.2.33
jmw5> route add default gw 192.168.2.33
```

Reaching the hosts on the 100 Mbps LAN and ppp link

Use of proxy arp on *jmw7* allows any host that can reach 192.168.2.0 to do so.

```
jmw7> cat arpon
echo "1" > /proc/sys/net/ipv4/ip_forward
/sbin/arp -v -s 192.168.2.35    00:00:77:8e:6d:b9 pub
/sbin/arp -v -s 192.168.2.66    00:00:77:8e:6d:b9 pub
```

The routing table on jmw7

```
 jmw7> /sbin/route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.2.66    0.0.0.0         255.255.255.255 UH    0      0        0 ppp0
192.168.2.32    0.0.0.0         255.255.255.224 U     0      0        0 eth0
192.168.2.0     0.0.0.0         255.255.255.224 U     0      0        0 lec0
127.0.0.0       0.0.0.0         255.0.0.0       U     0      0        0 lo
0.0.0.0         192.168.2.5     0.0.0.0         UG    0      0        0 lec0
 ==>
```

The arp cache on jmw7

```
 ==> /sbin/arp -n
Address                  HWtype  HWaddress           Flags Mask    Iface
192.168.2.4              ether   00:00:77:88:18:BC   C             lec0
192.168.2.5              ether   00:00:77:88:A4:95   C             lec0
192.168.2.6              ether   00:00:77:88:A1:15   C             lec0
192.168.2.35                     (incomplete)                      eth0
192.168.2.7              ether   00:00:77:88:A5:A5   C             lec0
192.168.2.1              ether   00:20:48:2E:00:EE   C             lec0
192.168.2.2              ether   00:00:77:97:C3:A5   C             lec0
130.127.48.184           *       *                   MP            lec0
192.168.2.66             *       *                   MP            lec0
192.168.2.35             *       *                   MP            lec0
```

The routing table on jmw

```
# netstat -nr

Routing Table: IPv4
  Destination           Gateway           Flags  Ref   Use   Interface
-------------------- -------------------- ----- ----- ------ ---------
192.168.2.0          130.127.48.113        UG       1    176
130.127.48.0         130.127.48.24         U        1   1269  hme0
224.0.0.0            130.127.48.24         U        1      0  hme0
default              130.127.48.1          UG       1    885
127.0.0.1            127.0.0.1             UH       2      8  lo0
```

The routing table on jmw7

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.2.34    *               255.255.255.255 UH    0      0        0 eth0
jmw7            *               255.255.255.255 UH    0      0        0 ppp0
192.168.2.32    *               255.255.255.224 U     0      0        0 eth0
127.0.0.0       *               255.0.0.0       U     0      0        0 lo
default         *               0.0.0.0         U     0      0        0 ppp0
```

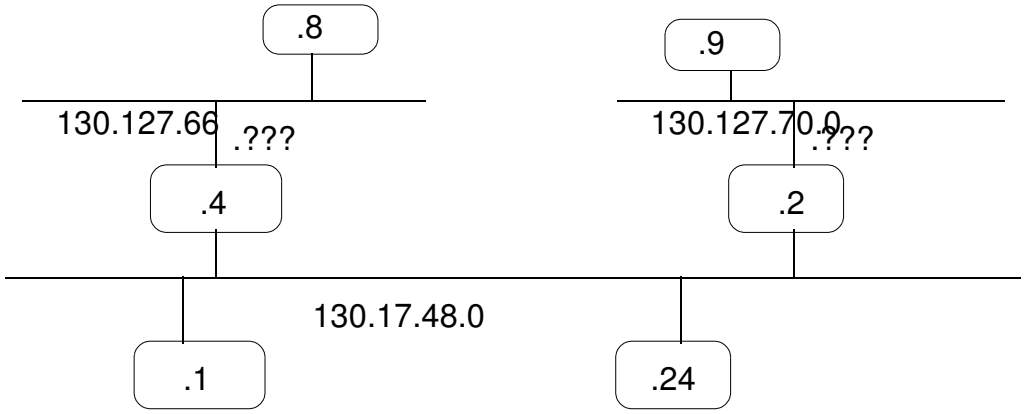
Exercise:

Arrive at a scheme for access to/from *jmw5* when it is attached to *jmw4*

ICMP Redirects

Objective:

If default router knows real router is on same LAN, a hop can be saved




```
jmw# netstat -rn
Routing tables
Destination       Gateway      Flags    Refcnt Use      Interface
130.127.66.32   130.127.48.4   UGHD     0      5           le0
130.127.66.8    130.127.48.4   UGHD     0      5           le0
130.127.70.9    130.127.48.2   UGHD     0      8           le0
127.0.0.1       127.0.0.1      UH       1      4111        lo0
130.127.66.2    130.127.48.4   UGHD     0      1           le0
default         130.127.48.1   UG       0      942         le0
130.127.48.0    130.127.48.24  U        27     1022954     le0
jmw#
```

ICMP Redirect Format

Type (5) | Code (0–3) | Checksum

New Destination Router Address

IP Header + 1st 8 bytes of original msg

Code values

0 – Redirect for network

1 – Redirect for host

2 – Redirect for TOS and network

3 – Redirect for TOS and host

Network type redirects

Can reduce the number of redirects sent and the size of routing tables.

But the books says DON'T use them in a subnetted environment as they can confuse some hosts

The 4.4BSD kernel generates an ICMP redirect i.f.f.

Outgoing interface = incoming interface
Outgoing route must not have been created by a redirect or be the default route
The datagram must not be source routed.

Receiver of an redirect should ensure

New router is directly connected.
Redirect must be *from* the current router for that destination
Specified new router can *not* be the original host!
Route must be indirect

Router discovery protocol

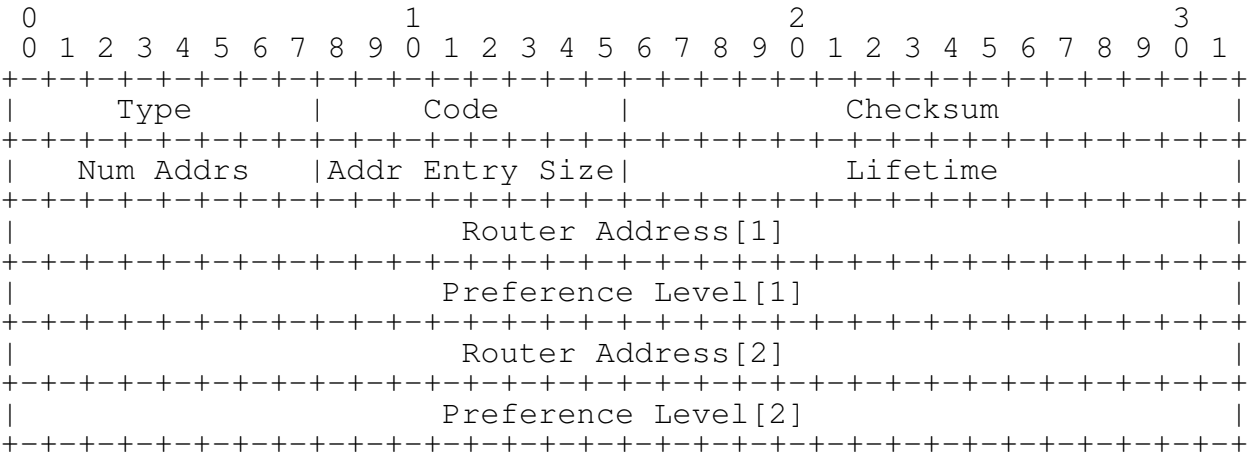
Specified in RFC 1256

Advertisements are generated by routers:
Periodically (450 – 600) seconds apart
In response to Solicitations

Solicitations are generated by hosts who have lost or don't know their default router.

Neither type of message is stored and forwarded.
==> Sender and receiver are always directly connected

3. Message Formats
ICMP Router Advertisement Message



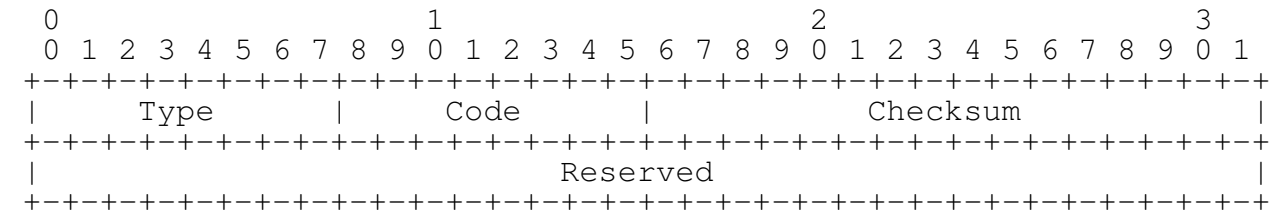
IP Header Fields:

Source Address	An IP address belonging to the interface from which this message is sent.
Destination Address	The configured AdvertisementAddress or the IP address of a neighboring host.
Time-to-Live	1 if the Destination Address is an IP multicast address; at least 1 otherwise.

ICMP Fields:

Type	9
Code	0
Checksum	The 16-bit one's complement of the one's complement sum of the ICMP message, starting with the ICMP Type. For computing the checksum, the Checksum field is set to 0.
Num Addrs	The number of router addresses advertised in this message.
Addr Entry Size	The number of 32-bit words of information per each router address (2, in the version of the protocol described here).
Lifetime	The maximum number of seconds that the router addresses may be considered valid.
Router Address[i], i = 1..Num Addrs	The sending router's IP address(es) on the interface from which this message is sent.
Preference Level[i], i = 1..Num Addrs	The preferability of each Router as a default router address, relative to other router addresses on the same subnet. A signed, twos-complement value; higher values mean more preferable.

ICMP Router Solicitation Message



IP Fields:

Source Address	An IP address belonging to the interface from which this message is sent, or 0.
Destination Address	The configured SolicitationAddress.
Time-to-Live	1 if the Destination Address is an IP multicast address; at least 1 otherwise.

ICMP Fields:

Type	10
Code	0
Checksum	The 16-bit one's complement of the one's complement sum of the ICMP message, starting with the ICMP Type. For computing the checksum, the Checksum field is set to 0.
Reserved	Sent as 0; ignored on reception.

4. Router Specification

4.1. Router Configuration Variables

A router that implements the ICMP router discovery messages must allow for the following variables to be configured by system management; default values are specified so as to make it unnecessary to configure any of these variables in many cases.

For each multicast interface:

AdvertisementAddress
: The IP destination address to be used for multicast Router Advertisements sent from the interface. The only permissible values are the all-systems multicast address, 224.0.0.1, or the limited-broadcast address, 255.255.255.255. (The all-systems address is preferred wherever possible, i.e., on any link where all listening hosts support IP multicast.)

  Default: 224.0.0.1 if the router supports IP multicast on the interface, else 255.255.255.255

MaxAdvertisementInterval
: The maximum time allowed between sending multicast Router Advertisements from the interface, in seconds. Must be no less than 4 seconds and no greater than 1800 seconds.

  Default: 600 seconds

MinAdvertisementInterval
: The minimum time allowed between sending unsolicited multicast Router Advertisements from the interface, in seconds. Must be no less than 3 seconds and no greater than MaxAdvertisementInterval.

  Default: 0.75 * MaxAdvertisementInterval

Connecting remotely via SLIP and PPP

The main problem:

Converting

The Connection (Terminal Emulation) Protocol to the Communication Protocol (Slip or PPP)

Standard dial−in procedure

Remote (Client)

Start terminal emulator
Go online
ATDT −−− etc
Connect

Host (server)

Getty variant answers
Forks execs login with stdin owning the line.

User at the remote site must reproduce the scenario above

Using the remote terminal only.

A three step procedure is used:

Convert ownership of the com port on the host to Slip or PPP
Exit the terminal emulator at the remot site *without hangin up.*
Convert ownership of the com port at the remote site to Slip or PPP.

Some automated procedures exist for doing this

Slip −>DIP
PPP −−> expect, chat

Setting up a *simple* PPP server in linux.

Create a user called *ppp*
Make the default shell */home/ppp/ppplogin*

```
#!/bin/sh
mesg n
stty -echo
exec /usr/sbin/pppd -detach silent
```

Dial the server using your favorite terminal emulator and login as *ppp*

In a separate (remote/home) linux session enter the command

```
/usr/sbin/pppd /dev/cua0 9600 130.127.48.250:130.127.48.249
```

An overview of the protocols

SLIP (RFC 1055)
Framing 0xC0 ––– data ––––– 0xCO
Transparency via a form of byte stuffing
Data 0xC0 sent as 0xDB, 0xDC
Data 0xDB sent as 0xDB, 0xDD

Shortcomings of SLIP
Need IP addresses for both ends of the link
Each end must MANUALLY configure the other's IP address (It can't be determined dynamically).
No error detection capability included.

CSLIP (aka Van Jacobson header compression) RFC 1144
Reduces TCP+IP header from 40 to 3 or 5 bytes

PPP (RFC 1548)

Framing – ISO HDLC

Flag –	0x7E
Addr –	0xFF
Cntl –	0x03
Protocol	0x0021 (IP Information)
	0xC021 (Link control)
	0x8021 (Network Control)
Data	
CRC	16 bit CCITT
Flag	0x7E

Transparency

Bit stuffing on synchronous links
Byte stuffing on ASYNC
0x7D => following byte is the "real" data but with 6th bit inverted)

Data 0x7E sent as 0x7D5E
Data 0x7D sent as 0x7D5D
Bytes less than 0x20 are all escaped.
0x01 sent as 0x21

Priniciple advantages of PPP over SLIP

Multiple protocols on a single serial line
CRC based frame check seqeunce
Dynamic IP address negotiation
Header compression (like CSLIP)
LCP for option negotiation –– like which bytes are escaped.