KUNGL TEKNISKA HÖGSKOLAN
Institutionen för teleinformatik
CCSlab

2G1305 Internetworking/Internetteknik
Winter 2002, Period 3

Module 2: IP Routing

Lecture notes of G. Q. Maguire Jr.

For use in conjunction with *TCP/IP Illustrated*, volumes 1 and 3, by W. Richard Stevens and *TCP/IP: Principles, Protocols, and Architectures, Vol. 1*, by Douglas E. Comer, Prentice Hall, 4th ed. 2000.

Last modified: 2002.01.20:20:50

Lecture 2: IP Routing Outline

- Useful Diagnostic Tools
- ARP, ICMP, Ping, Traceroute
- IP Routing
- Dynamic Routing Protocols

Tools Used: tcpdump Program

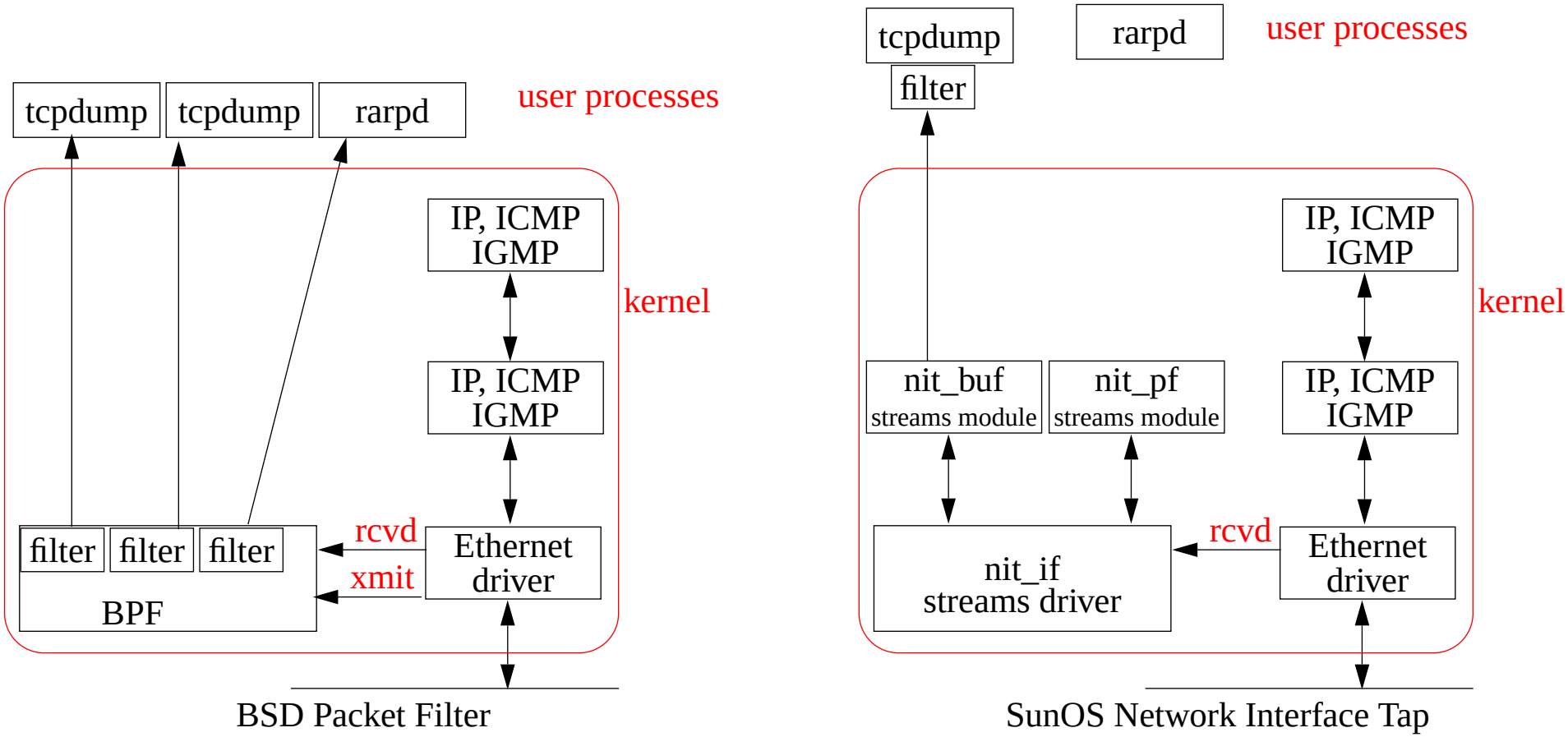
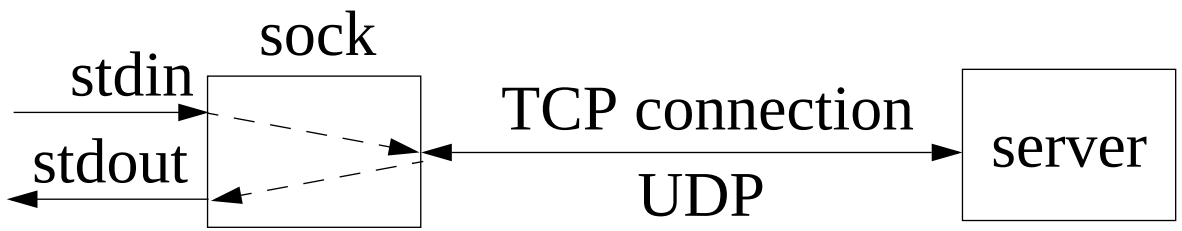


Figure 18: Two alternatives to get packets

Note the BSF packet filter gets a copy of both the received and transmitted packets.

Tools Used: sock Program

- A simple test program to generate TCP, UDP data
- To test and debug TCP, UDP implementations



- Interactive client: default
- Interactive server: -s
- Source client: -i
- Sink server: -i -s
- Default TCP, -u for UDP

Source Code Available: (Tcpdump and sock)

For Win95/98/2000/NT: *http://netgroup-serv.polito.it/windump/*

For BSD alike: *ftp://ftp.uu.net/published/books/stevens.tcpipiv1.tar.Z*

Tools Used: sock + tcpdump

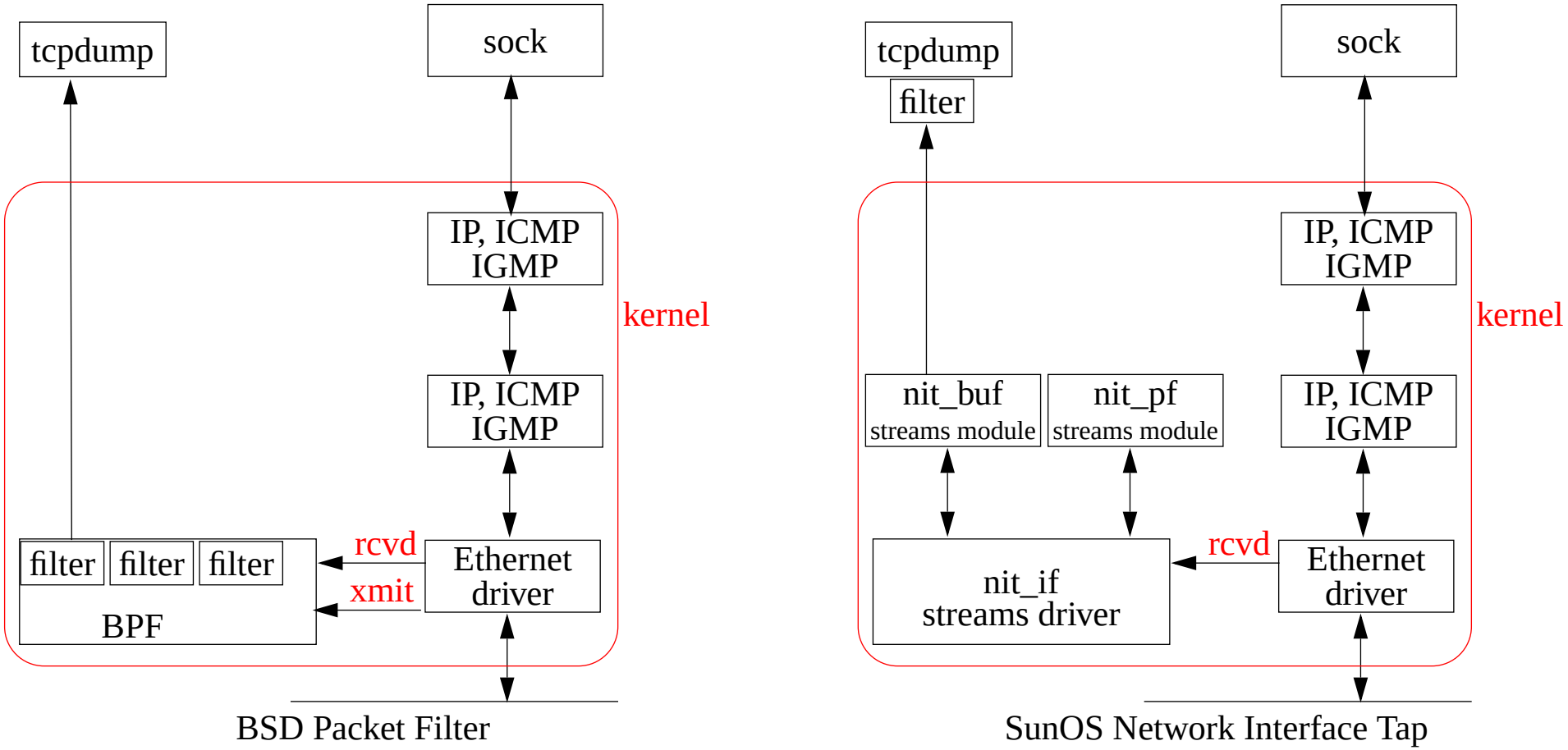


Figure 19: Two alternatives to generate and dump packets

What to do with a new computer?

We will assume that the computer has an ethernet interface:

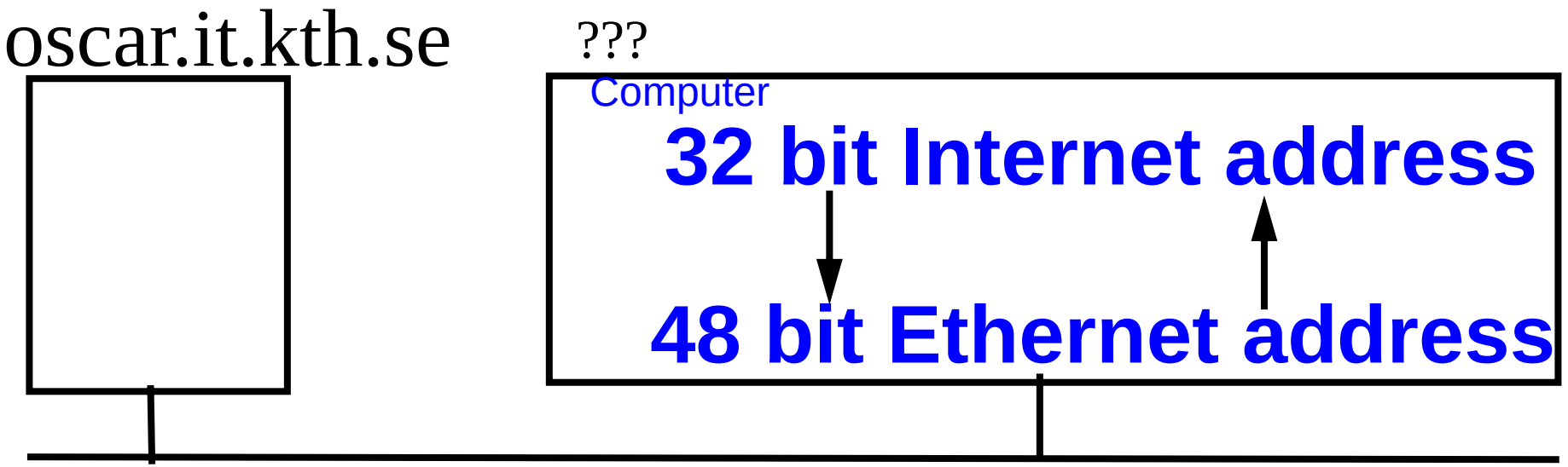


Figure 20: Name and IP Address needed!

- Direct mapping - requires no I/O, just a computation; hard to maintain; and requires stable storage (since you have to store the mappings somewhere) *or*
- Dynamic Binding - easier to maintain; but has a delay while messages are exchanged

Address Resolution: ARP, RARP

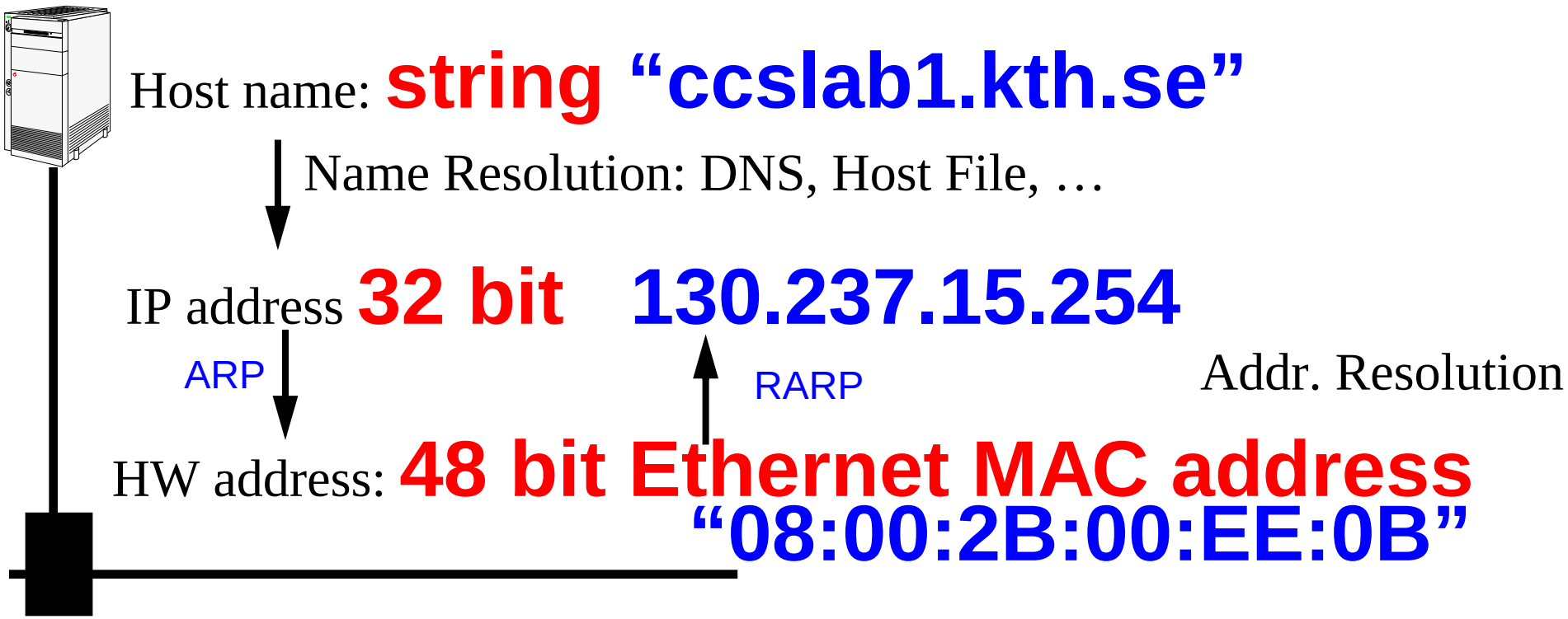


Figure 21: mapping between host names, IP address, and MAC address

ARP - Address Resolution Protocol

RARP- Reverse ARP

ARP ≡ Address Resolution Protocol (RFC826)

Address Resolution Protocol (ARP) - allows a host to find the physical address of a target host **on the same network**, given only the target's internet address.

- Sending host (source) wants to send an IP datagram, but does not know the corresponding ethernet address
- ARP request - broadcast to every host on the network (i.e., EtherDST=0xFFFFFF), TYPE=0x0806
- Destination host: "It is my address!" and sends an ARP reply
- Source host - receives the ARP reply, and now uses it to send the IP datagram

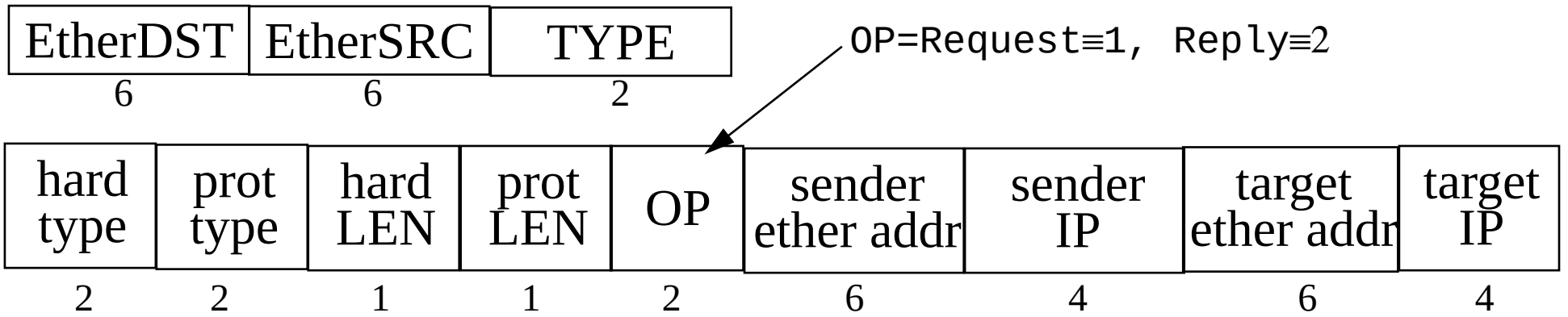


Figure 22: Format of ARP request/reply packet (see Stevens, Vol. 1, figure 4.3, pg. 56)

ARP example 1

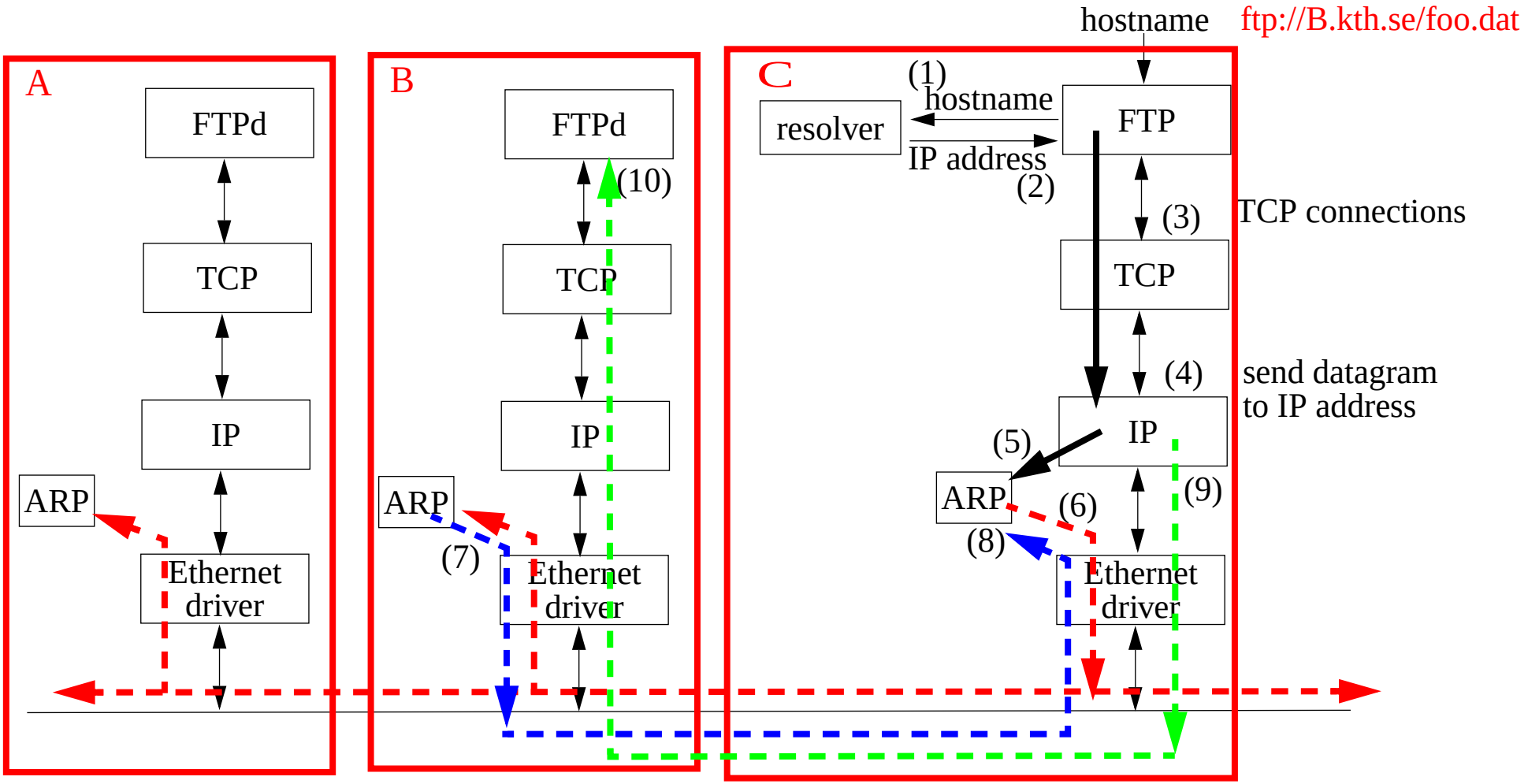


Figure 23: Using ARP to determine MAC address of B

Address Resolution Cache

Since you have just looked up the address, save it for reuse:

- to limit ARP traffic
- works because of correlations in use of addresses

You can examine the arp cache:

```
arp -a
machine-name (x.x.x.x) at xx:xx:xx:xx:xx:xx
…

arp -an
(x.x.x.x) at xx:xx:xx:xx:xx:xx
…
```

Note that the later form with the “n” option does *not* lookup the hostname, this is *very useful* when you don’t yet have a name resolution service working!

ARP Refinements

Since the sender’s Internet-to-Physical address binding is in every ARP broadcast; (all) receivers update their caches before processing an ARP packet

ARP Timeouts

- If there is no reply to an ARP request
  - the machine is down or not responding
  - request was lost, therefore retry (but not too often)
  - eventually give up (When?)
- ARP cache timeouts
  - Berkeley implementations timeout
    - completed entry in 20 minutes
    - incomplete entry in 3 minutes
  - Linux:
    - for entries to which there has been no traffic a timeout occurs at `gc_stale_time`, set to 1 minute by default
  - Microsoft Windows NT
    - Using the registry editor, see HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters enter ArpCacheLife default value is set to 2 minutes
  - Cisco IOS v10.0 and above
    - select interface then “arp timeout xxxx”, default value is set to 240 minutes (14400 seconds) and can be changed on a per-interface basis
  - RFC 1533: DHCP Options and BOOTP Vendor Extensions, defines:
    - ARP Cache Timeout Option (code for this option is 35).
  - Host Requirements RFC - says entries should be timed out **even if in use**

ARP example 2

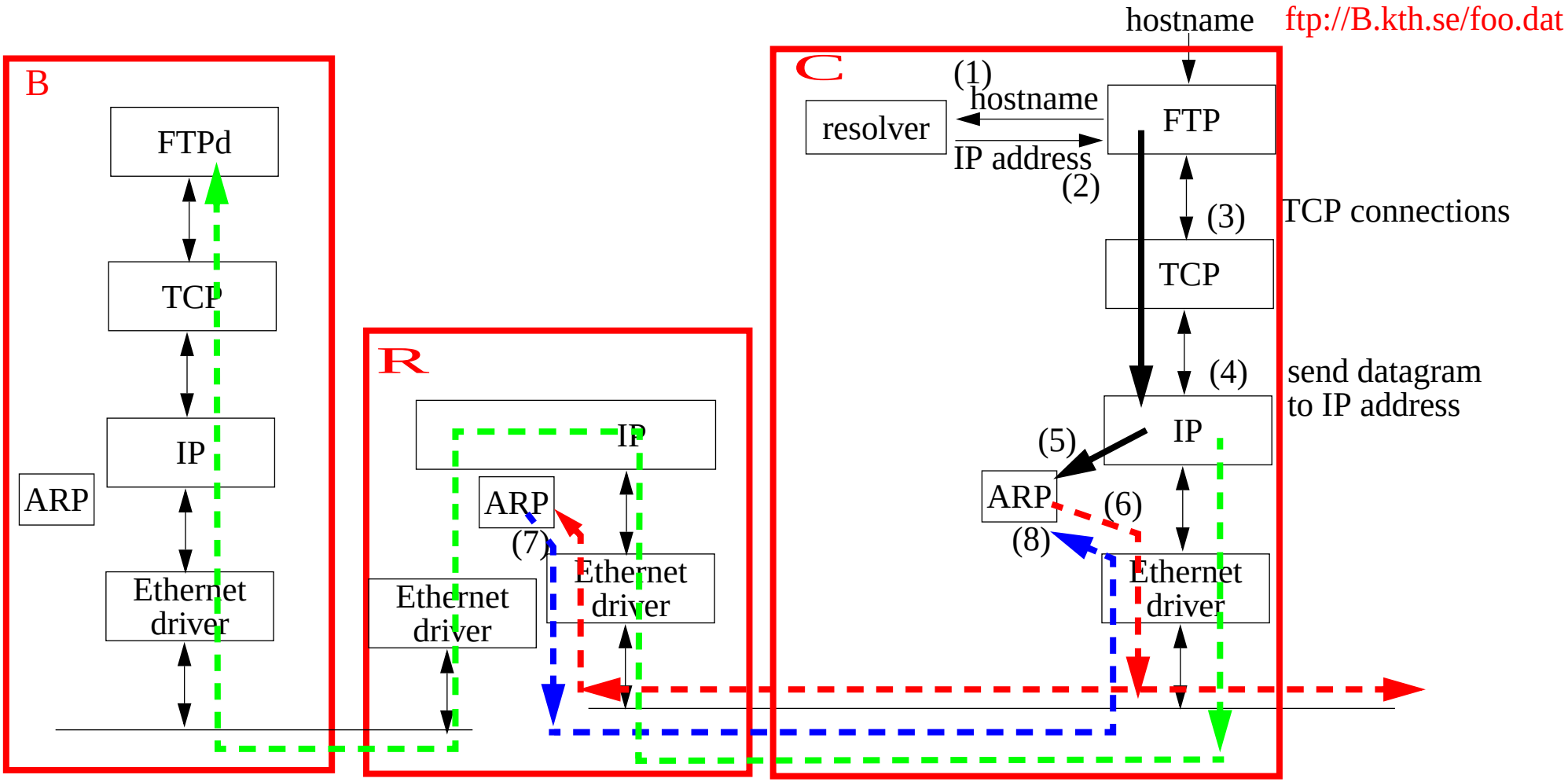


Figure 24: Router (R) doing a Proxy ARP to provide MAC address of B

Proxy ARP (RFC 826)

Lets a **router** on the network answer for a host which is NOT necessarily on the local network segment.

But how does this router know?

- It can make an ARP request itself or
- Perhaps it already knows - because it has an entry in it's ARP cache

For an example of using proxy arp with subnetting see:

- *http://www.linuxdoc.org/HOWTO/mini/Proxy-ARP-Subnet/why.html* and
- *http://www.linuxdoc.org/HOWTO/mini/Proxy-ARP-Subnet/how.html*

Gratuitous ARP

Host sends a request for its own address

- generally done at boot time to inform other machines of its address (possibly a new address) - they get a chance to update their cache entries immediately
- lets hosts check to see if there is another machine claiming the same address ⇒ “duplicate IP address sent from Ethernet address a:b:c:d:e:f”

As noted before, hosts have paid the price by servicing the broadcast, so they can cache this information - this is one of the ways the proxy ARP server could know the mapping.

Note that faking that you are another machine can be used to provide failover for servers (see for example heartbeat, fake, etc. at *http://www.linux-ha.org/download/* for a send_arp program).

Additional ARP commands

- publish entries (i.e., mechanically make an entry and answer replies)

Publishing entires is one way that (embedded) devices can learn their IP addres.

```
# arp -s birkexample 08:00:2B:00:EE:0B pub
# arp -an
 (192.168.1.1) at 0:4:5a:de:e8:f9 ether
  …
 (172.16.32.20) at 8:0:2b:0:ee:b ether permanent published
```

where `birkexample` has the IP address: 172.16.32.20

- expliclity delete entries

```
# arp -d birkexample
birkexample (172.16.32.20) deleted
# arp -an
 (192.168.1.1) at 0:4:5a:de:e8:f9 ether
```

non ARP example 1

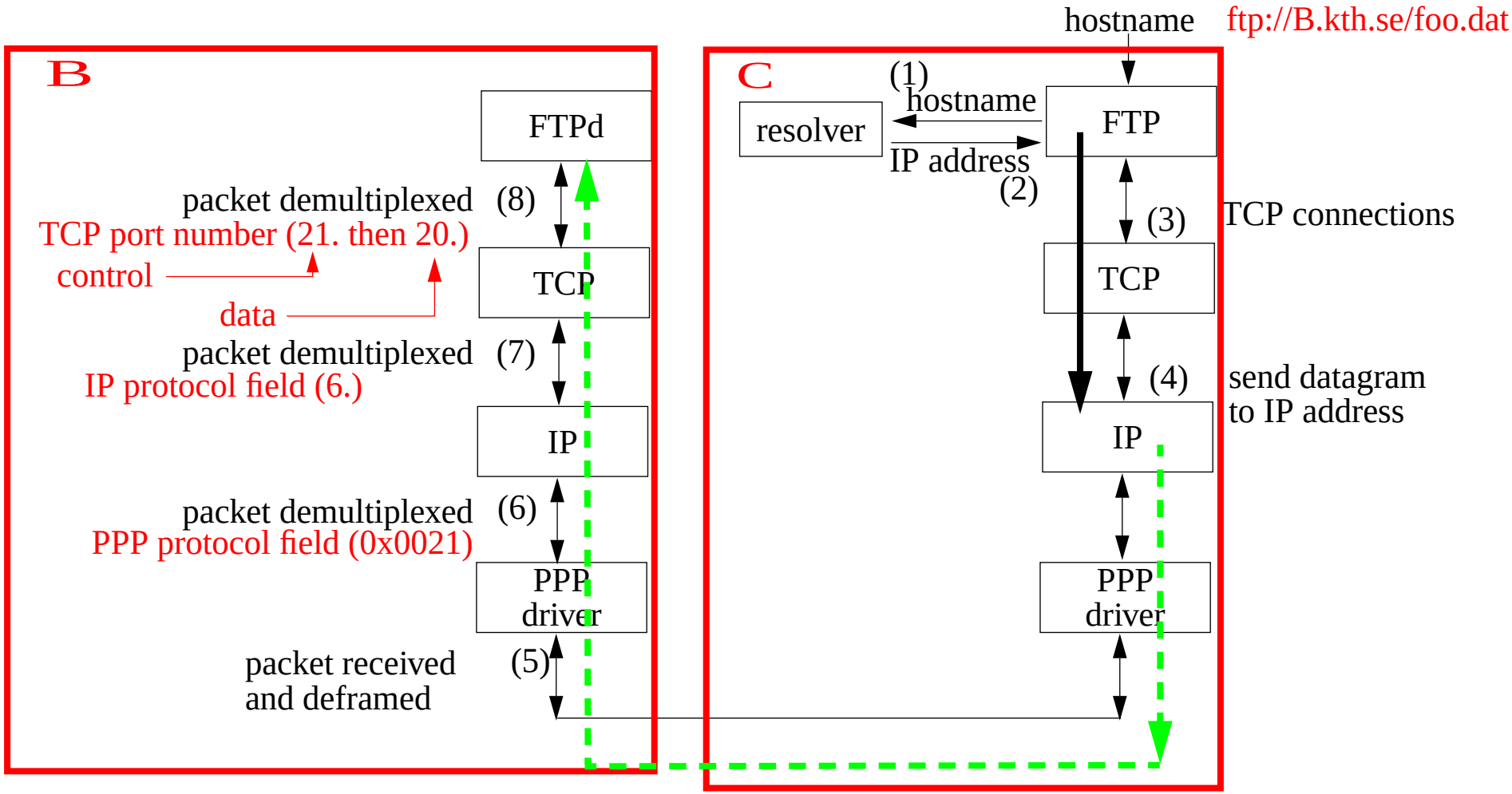


Figure 25: On a point-to-point link there is no need for ARP (figure also shows explicitly the demultiplexing)

Note that the PPP protocol field plays a role similar to the ethernet frame type.

RARP: Reverse Address Resolution Protocol (RFC 903)

How to get you own IP address, when all you know is your link address.

- Necessary if you don't have a disk or other stable store
- RARP request - broadcast to every host on the network (i.e., EtherDST=0xFFFFFF), TYPE=0x8035
- RARP server: "I know that address!" and sends an RARP reply
- Source host - receives the RARP reply, and now knows its own IP addr

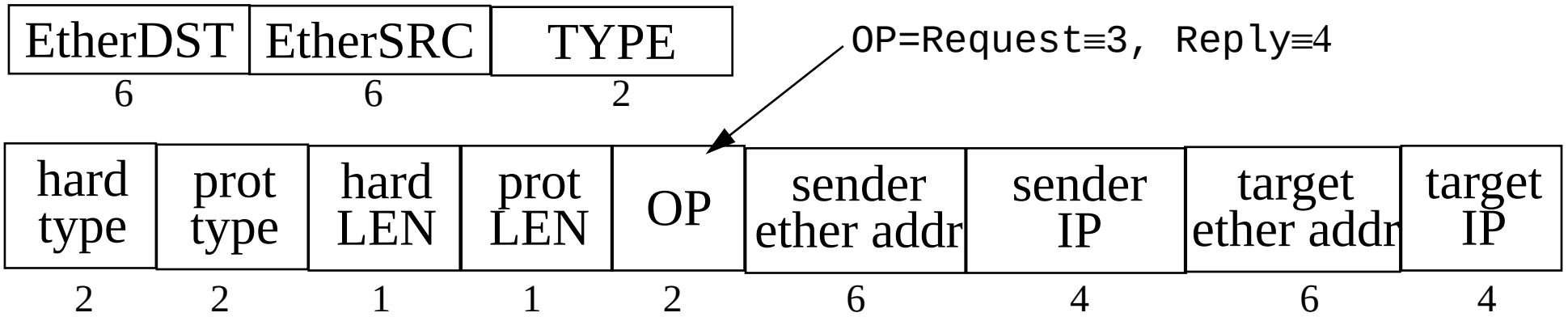


Figure 26: Format of RARP request/reply packet

You can now see what the "publish" aspect of the `arp` command is for.

RARP server

Someone has to know the mappings - quite often this is in “/etc/ethers”

Since this information is generally in a file, RARP servers are generally implemented as **user processes.**

- Unlike ARP responses which are generally part of the TCP/IP implementation (often part of the kernel).
- How does the process get the packets - since they aren’t IP and won’t come across a socket?
  - BSD Packet filters
  - SVR4 Data Link Provider Interface (DLPI)
  - SUN’s Network Interface Tap (NIT)
  - Interestingly in the appendix to RFC 903 an alternative to having data link level access was to have two IOCTLs, one that would "sleep until there is a translation to be done, then pass the request out to the user process"; the other means: "enter this translation into the kernel table"
- RARP requests are sent as hardware level broadcasts - therefore are not forwarded across routers:
  - multiple servers per segement - so in case one is down; the first response is used
  - having the router answer

Internet Control Message Protocol (ICMP)

ICMP is part of the same level as IP, but uses IP for transfers! ICMP is used by layer 3 entities to communicate with each other.

- ICMP PDU: type (8 bits); code (8 bits); checksum (16 bits); parameters (n*32 bits); information (variable length)
  for errors: the information field always includes the first 64 bits of the data field of the original datagram which caused the ICMP message
- ICMP messages include:
  - Destination Unreachable (Network/Host/Protocol/Port/…)
  - Time Exceeded (TTL expired)
  - Parameter problem - IP header error
  - Source Quench (requests source to decrease its data rate)
  - Redirect - tell source to send its messages to a “better address”
  - Echo Request/ Echo reply - for testing (e.g., “ping” program sends an Echo request)
  - Timestamp Request/ Timestamp reply
  - Information Request / Information reply
  - Address Mask Request / Reply
  - Traceroute
  - Datagram conversion error
  - Mobile Host Redirect/Registration Request/Registration Reply
  - IPv6 Where-Are-You/I-Am-Here

ICMP Port Unreachable Error

Example: (Stevens, Vol. 1, Section 6.5, pp. 77-78)

```
bsdi% tftp
tftp> connect svr4 888   specify host and port number
tftp> get temp.foo       try to fetch a file
Transfer times out.      about 25s later
tftp> quit
```

tcpdump output

1	0.0		arp who-has svr4 tell bsdi
2	0.002050	(0.0020)	arp reply svr4 is-at 0:0:c0:c2:9b:26
3	0.002723	(0.0007)	bsdi.2924 > svr4.8888 udp 20
4	0.006399	(0.0037)	svr4 > bsdi: icmp: svr4 udp port 8888 unreachable
5	5.000776	(4.9944)	bsdi.2924 > svr4.8888 udp 20
6	5.004304	(0.0035)	svr4 > bsdi: icmp: svr4 udp port 8888 unreachable
…			repeats every 5 seconds
11	20.001177	(4.9966)	bsdi.2924 > svr4.8888 udp 20
12	20.004759	(0.0036)	svr4 > bsdi: icmp: svr4 udp port 8888 unreachable

PING: Packet InterNet Groper or sonar echo

Ping was written by Mike Muuss[1] to test host reach-ability. Uses ICMP,most IP implementations support Ping server. Sends an ICMP echo request to a host

Format of ICMP message for Echo request/reply (see Stevens, Vol. 1, figure 7.1, pg. 86)

Type (0 or 8)	code (0)	16 bit checksum
16 bit identifier		16 bit sequence number
Optional data		

Look at ping across different connections[2]:

- LAN
- WAN
- Hardwired SLIP
- Dialup SLIP - extra delay due to the modems and the correction/compression

With IP record route (RR) option tracing the route of the ping datagram.

---

1. Mike Muuss was killed in an automobile accident on November 20, 2000. *http://ftp.arl.mil/~mike/*

2. For examples, see Stevens, Vol. 1, Chapter 7, pp. 86-90.

PING examples

On a Solaris machine:

bash-2.03$ /usr/sbin/ping cyklop.nada.kth.se *from a machine at IMIT*

cyklop.nada.kth.se is alive

bash-2.03$ /usr/sbin/ping -s cyklop.nada.kth.se

PING cyklop.nada.kth.se: 56 data bytes
64 bytes from cyklop.nada.kth.se (130.237.222.71): icmp_seq=0. time=3. ms
64 bytes from cyklop.nada.kth.se (130.237.222.71): icmp_seq=1. time=1. ms
64 bytes from cyklop.nada.kth.se (130.237.222.71): icmp_seq=2. time=1. ms
^C
----cyklop.nada.kth.se PING Statistics----
3 packets transmitted, 3 packets received, 0% packet loss
round-trip (ms) min/avg/max = 1/1/3

Why did the first ping take longer?

On a HP-UX 11.0 machine:

ping -ov www.kth.se *from a machine on Telia's ADSL network*

```
PING www.kth.se: 64 byte packets
64 bytes from 130.237.32.51: icmp_seq=0. time=54. ms
64 bytes from 130.237.32.51: icmp_seq=1. time=38. ms
64 bytes from 130.237.32.51: icmp_seq=2. time=11. ms
64 bytes from 130.237.32.51: icmp_seq=3. time=11. ms
64 bytes from 130.237.32.51: icmp_seq=4. time=11. ms
^C
----www.kth.se PING Statistics----
5 packets transmitted, 5 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 11/25/54
5 packets sent via:                    this is based on the record route information (caused by "-ov")
    217.208.194.247 - fls31o268.telia.com
    213.64.62.150   - fre-d4-geth6-0.se.telia.net
    213.64.62.154   - fre-c3-geth6-0.se.telia.net
    195.67.220.1    - fre-b1-pos0-1.se.telia.net
    130.242.94.4    - STK-PR-2-SRP5.sunet.se
    130.242.204.130 - STK-BB-2-POS4-3.sunet.se
    130.242.204.121 - stockholm-1-FE1-1-0.sunet.se
    130.237.32.3    - [ name lookup failed ]
    130.237.32.51   - oberon.admin.kth.se
```

Novel IPX/SPX Addresses

Another approach to addresses

IPX/SPX == INternetwork Packet Exchange/Sequenced Packet Exchange

IPX address: 32 bits of network ID and 48 bits of host ID (the ethernet address)

Problems:

- There is no central authority for allocating the network IDs
  - ✘ So if you interconnect multiple IPX networks you may have to renumber every network
- If you change ethernet cards, you get a new address!
- Assumes that all machines are attached to a high capacity LAN.

Advantages

- You only have to assign network numbers, then the hosts figure out their own address. Simpler administration.

Novell NetWare provides: Service Advertising Protocol (SAP), Routing Information Protocol (RIP), and NetWare Core Protocol (NCP).

ISP Backbone Networks

A good summary of Internet backbones in “Is the Internet in trouble?” by Robin Gareiss, Data Communications, Sept. 21, 1997, pp. 36-50. Pg. 44 gives a list of URL’s for ISP backbones.

Boardwatch magazine - a publication for ISPs has some network backbone maps (*http://boardwatch.internet.com/*)

Useful Tool: Traceroute Programs

Developed by Van Jacobson to see the route that IP datagrams follow from one host to another. Traceroute uses ICMP, TTL field and an *unreachable UDP port*.

```
svr % traceroute slip
traceroute to slip (140.252.13.65), 30 hops max, 40 byte packets
1 bsdi (140.252.13.35) 20 ms 10 ms 10 ms          20 ms due to ARP
2 slip (140.252.13.65) 120 ms 120 ms 120 ms
```

tcpdump output

1	0.0		arp who-has bsdi tell svr4	
2	0.000586	(0.0006)	arp reply bsdi is-at 0:0:c0:6f:2d:40	
3	0.003067	(0.0025)	svr4.42804 > slip.33435 udp 12 [ttl 1]	
4	0.004325	(0.0013)	bsdi > svr4: icmp: time exceeded in-transit	
5	0.069810	(0.0655)	svr4.42804 > slip.33436 udp 12 [ttl 1]	
6	0.071149	(0.0013)	bsdi > svr4: icmp: time exceeded in-transit	
7	0.085162	(0.0140)	svr4.42804 > slip.33437 udp 12 [ttl 1]	
8	0.086375	(0.0012)	bsdi > svr4: icmp: time exceeded in-transit	
9	0.118608	(0.0322)	svr4.42804 > slip.33438 udp 12	*ttl=2*
10	0.226464	(0.1079)	slip > svr4: icmp: slip udp port 33438 unreachable	
11	0.287296	(0.0608)	svr4.42804 > slip.33439 udp 12	*ttl=2*
12	0.395230	(0.1079)	slip > svr4: icmp: slip udp port 33439 unreachable	
13	0.409504	(0.0608)	svr4.42804 > slip.33440 udp 12	*ttl=2*
14	0.517430	(0.1079)	slip > svr4: icmp: slip udp port 33440 unreachable	

tcpdump

Under HP-UX 11.0

```
# ./tcpdump -i /dev/dlpi0
tcpdump: listening on /dev/dlpi0
```

22:25:43.217866 birk2.5900 > nucmed35.50251: . ack 3089200293 win 8080 (DF)

22:25:43.290636 birk2.5900 > nucmed35.50251: P 0:4(4) ack 1 win 8080 (DF)

22:25:43.360064 nucmed35.50251 > birk2.5900: . ack 4 win 32768

22:25:43.363786 birk2.5900 > nucmed35.50251: P 4:167(163) ack 1 win 8080 (DF)

22:25:43.364159 nucmed35.50251 > birk2.5900: P 1:11(10) ack 167 win 32768

22:25:43.543867 birk2.5900 > nucmed35.50251: . ack 11 win 8070 (DF)

22:25:43.577483 birk2.5900 > nucmed35.50251: P 167:171(4) ack 11 win 8070 (DF)

22:25:43.640052 nucmed35.50251 > birk2.5900: . ack 171 win 32768

22:25:43.643793 birk2.5900 > nucmed35.50251: P 171:334(163) ack 11 win 8070 (DF)

22:25:43.644132 nucmed35.50251 > birk2.5900: P 11:21(10) ack 334 win 32768

22:25:43.750062 birk2.5900 > nucmed35.50251: . ack 21 win 8060 (DF)

22:25:43.873349 birk2.5900 > nucmed35.50251: P 334:338(4) ack 21 win 8060 (DF)

22:25:43.940073 nucmed35.50251 > birk2.5900: . ack 338 win 32768

13 packets received by filter

0 packets dropped by kernel

Routing

The internet protocols are based on moving packets from a source to a destination with each hop making a routing decision.

Two components to routing:

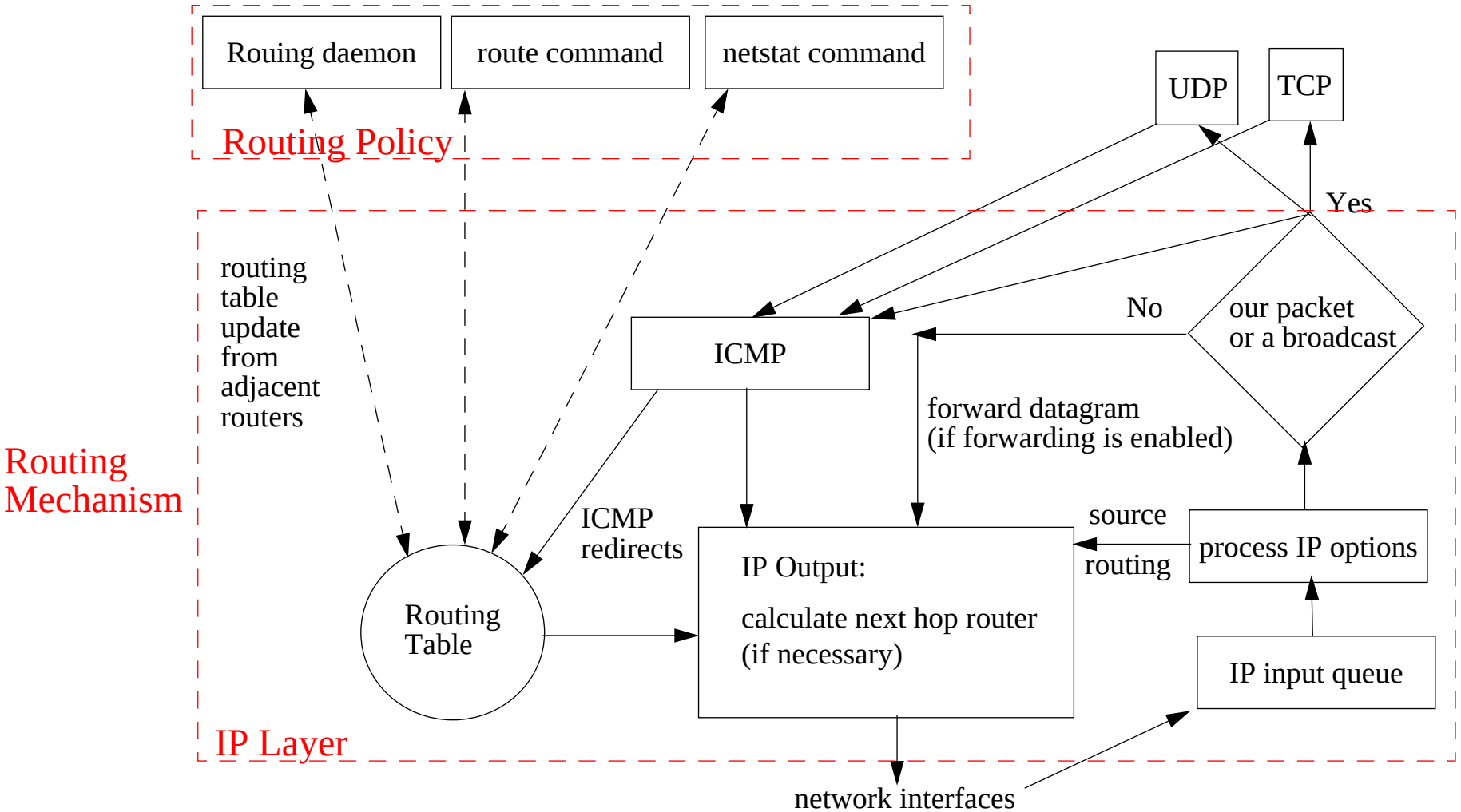
- packet forwarding - Routing Mechanism: search the routing table and decide which interface to send a packet out.
  - A matching host address? If no,
  - A matching network address? (using longest match) If no,
  - Default entry.
- computing routes - Routing Policy: rules that decide which routes should be added into the routing table.

Traditionally most of the complexity was in the later (i.e., computing routes) while packet forwarding was very straight forward -- this is no longer true due to QoS.

Routers vs. hosts -- a node can be both

- Routers forward IP packets
- Hosts generate or sink IP packets

Processing


Rouing daemon
route command
netstat command
Routing Policy
UDP
TCP
Yes
routing
table
update
from
adjacent
routers
No
our packet
or a broadcast
ICMP
forward datagram
(if forwarding is enabled)
Routing
Mechanism
ICMP
redirects
source
routing
process IP options
IP Output:
calculate next hop router
(if necessary)
Routing
Table
IP input queue
IP Layer
network interfaces

Combining layers

Many devices now combine processing of several layers:

- Switch/Routers: combine layer 2+3

Devices combining layers 3+4 are appearing - which extract “flows” based on looking at transport layer port numbers in addition to network addresses.

Basic Ethernet + IP Software Architecture

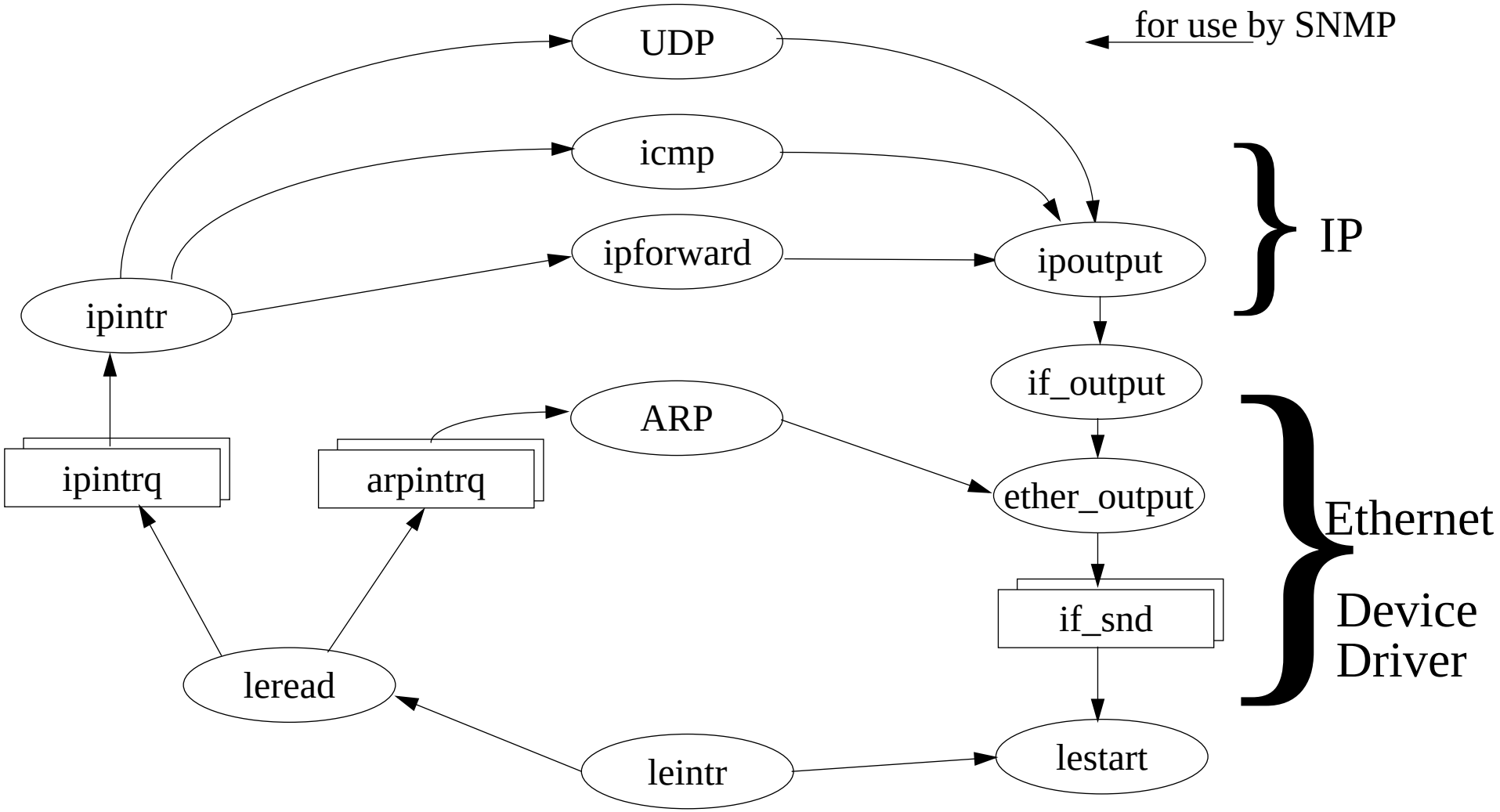


Figure 27: Basic Ethernet + IP Architecture - based on Stevens, TCP/IP Illustrated, Volume2

Basic Router Software Architecture

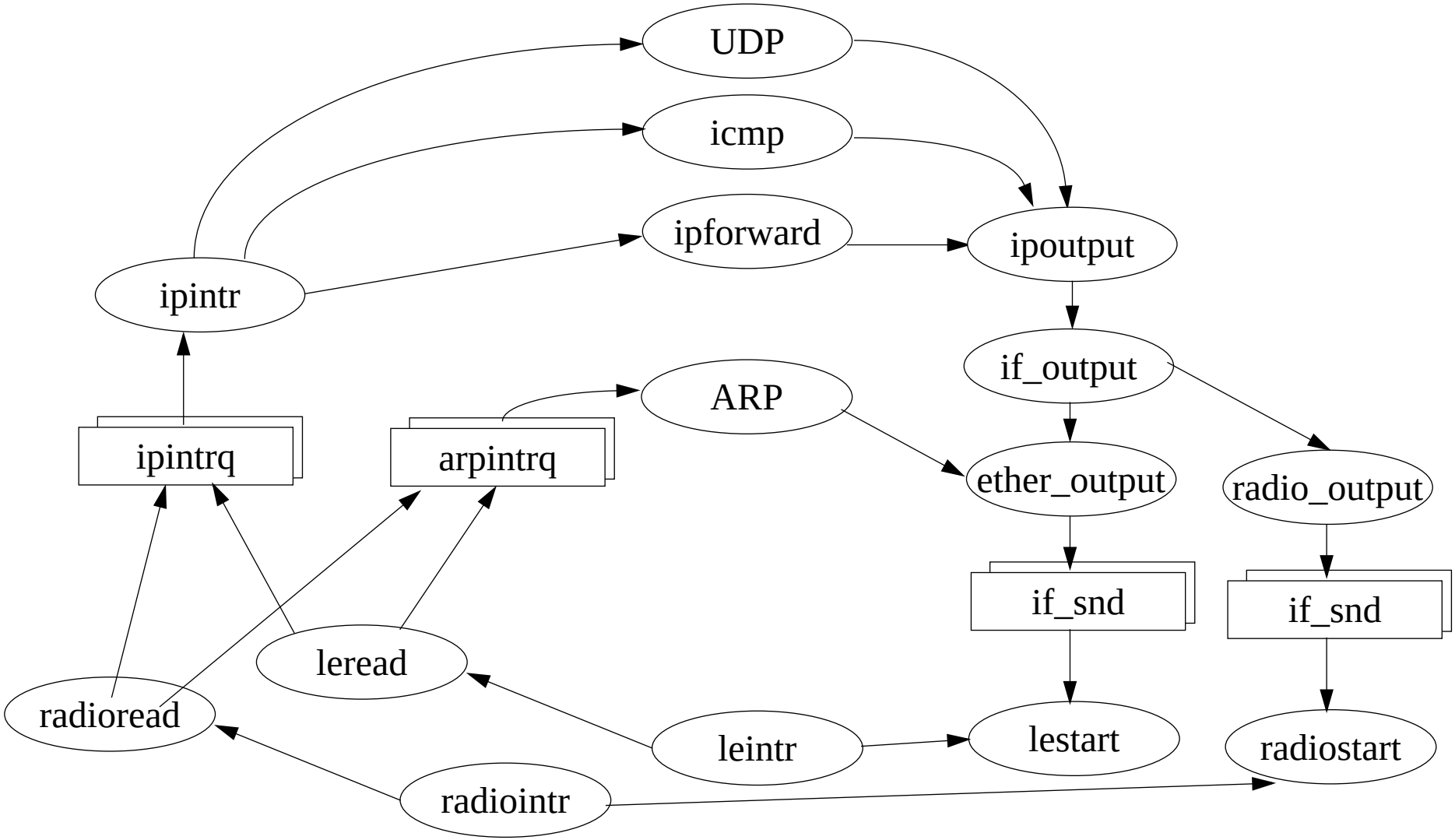


Figure 28: Basic Router Architecture (assuming a radio and an ethernet interface)

Basic Paths

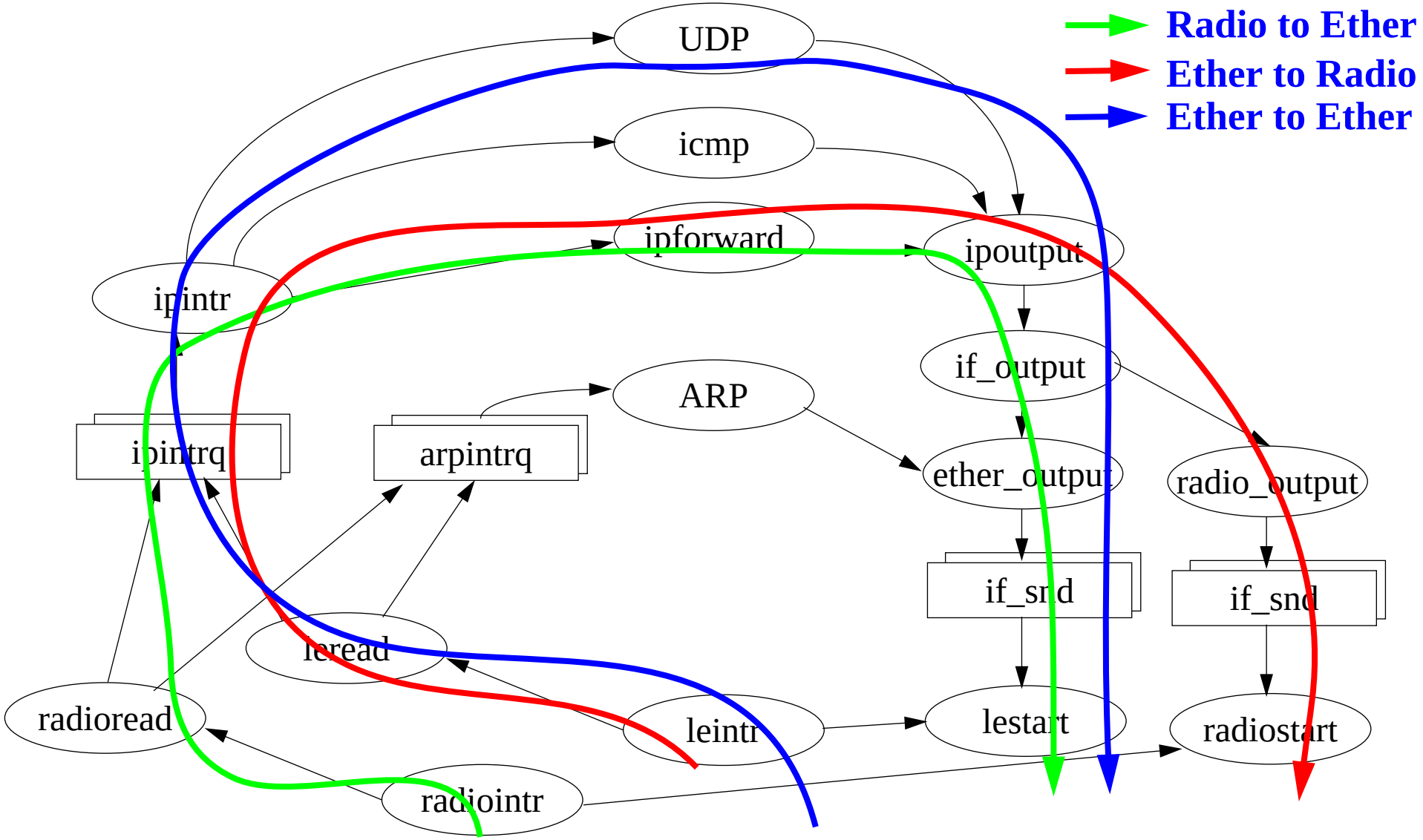


Figure 29: Basic Paths

Why is there Ethernet to Ethernet Communication?

Ethernet to Ethernet communication occurs to support SNMP, RMON, … - i.e., remote management and statistics.

Otherwise for a two port device, the two input to output paths (red and green) can be independent. Having independent paths could mean that the memories can be simpler, management of the queues becomes simpler, interrupts become simpler (since you don’t strictly need them as the state machines can run largely independently), etc.

Basic Bridging Paths

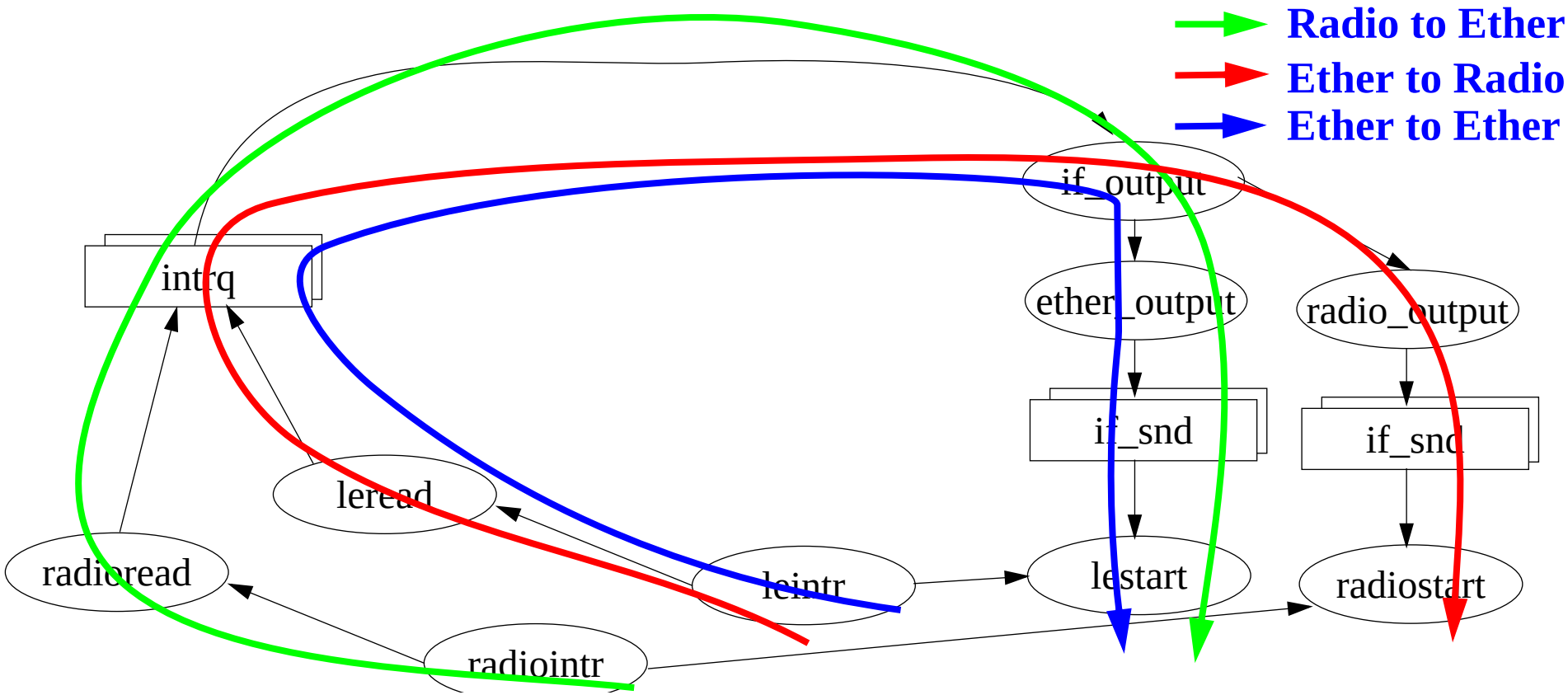


Figure 30: Basic Router Architecture

We don’t need network layer protocol processing to just perform bridging (but then of course we can’t support SNMP, RMON, … for maintenance, control, and statistics gathering - since we don’t have any higher layer protocols).

Interesting Bridging Paths

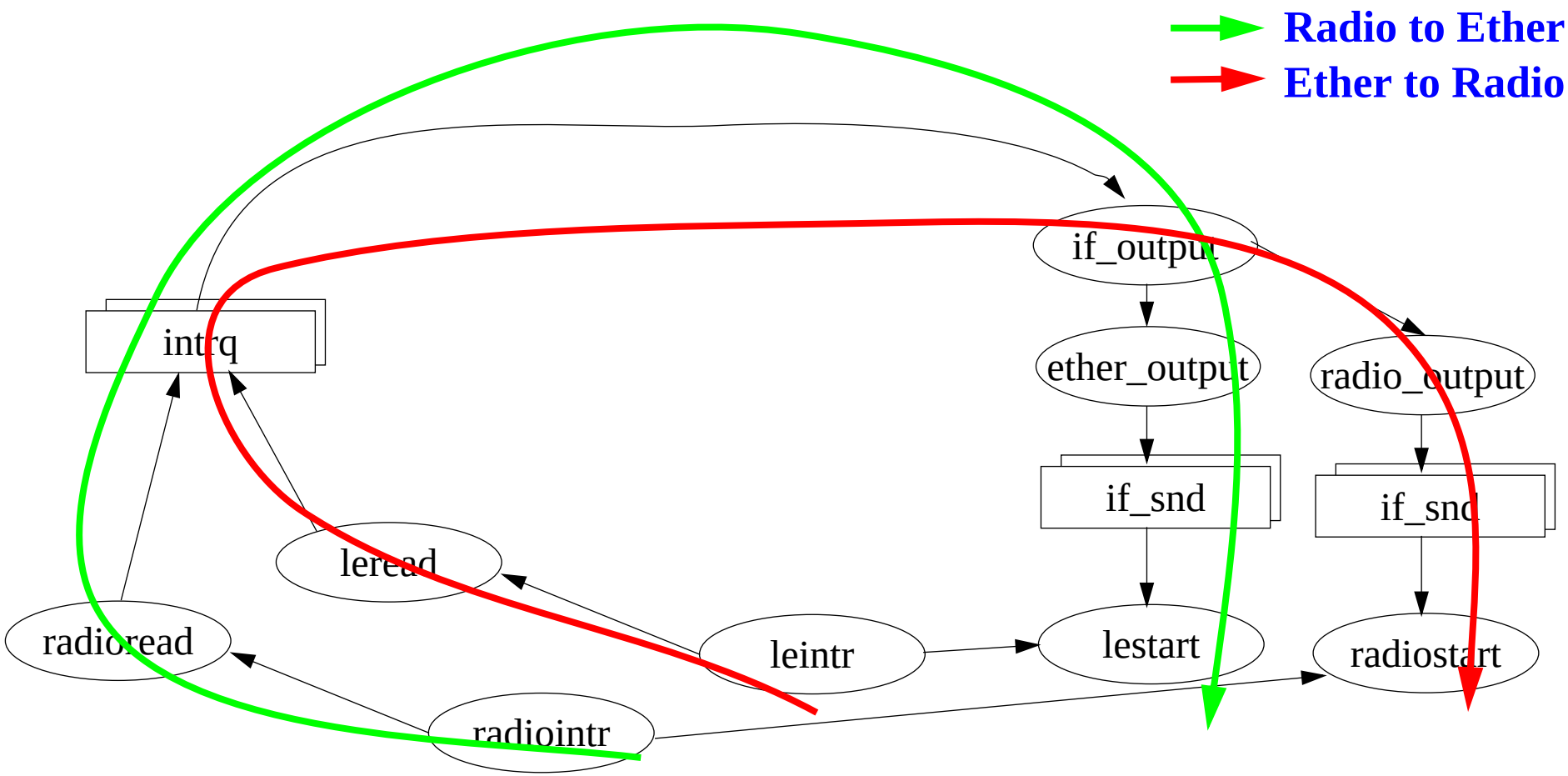


Figure 31: Interesting Bridging Paths

Since the Ethernet to Ethernet bridging path is not interesting for a pure bridge, we can eliminte this path.

Reduced Bridging Paths

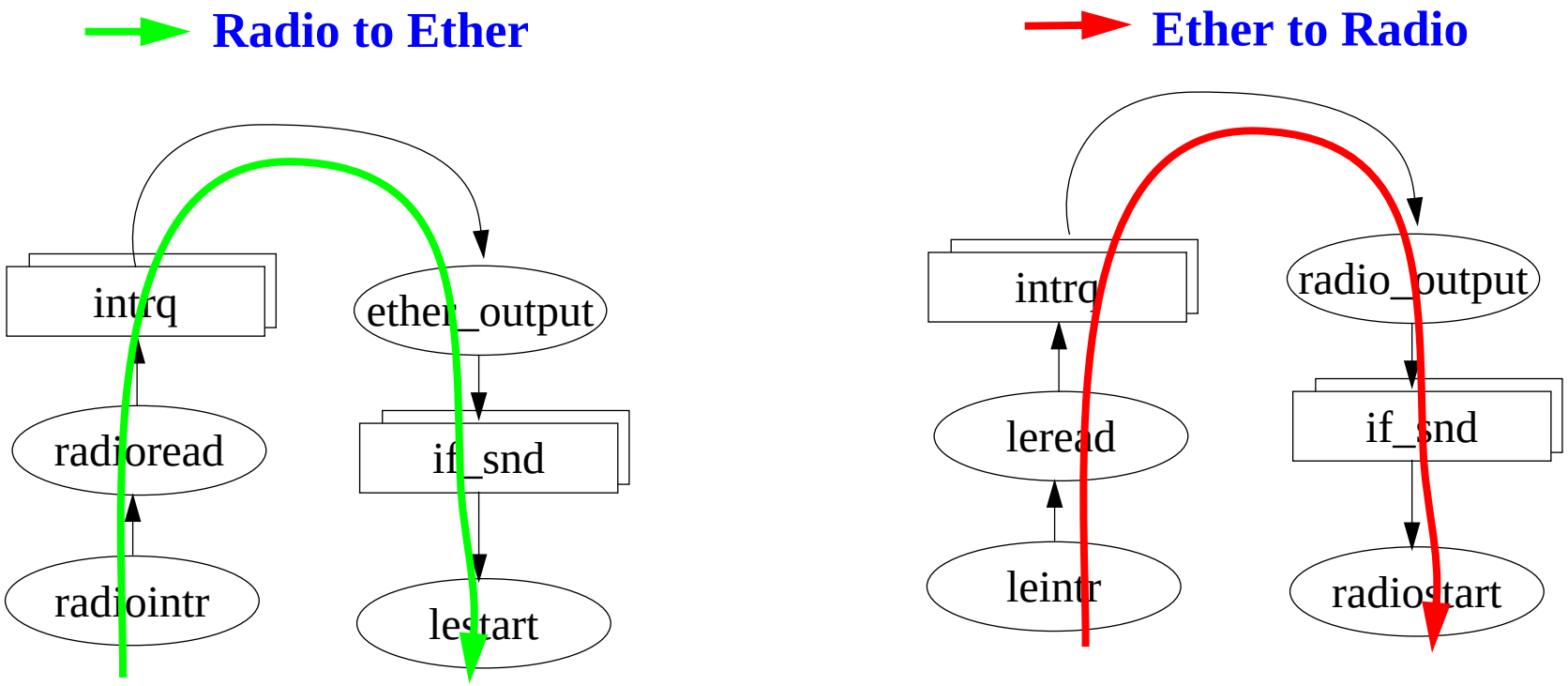


Figure 32: Reduced Bridging Paths

Since **frames** are only going either from the radio to the ethernet or from the ethernet to the radio, we can split these into two different machines. The only interaction in both cases will be at the MAC layer of each media, otherwise the paths are independent.

Frame In - Frame Out

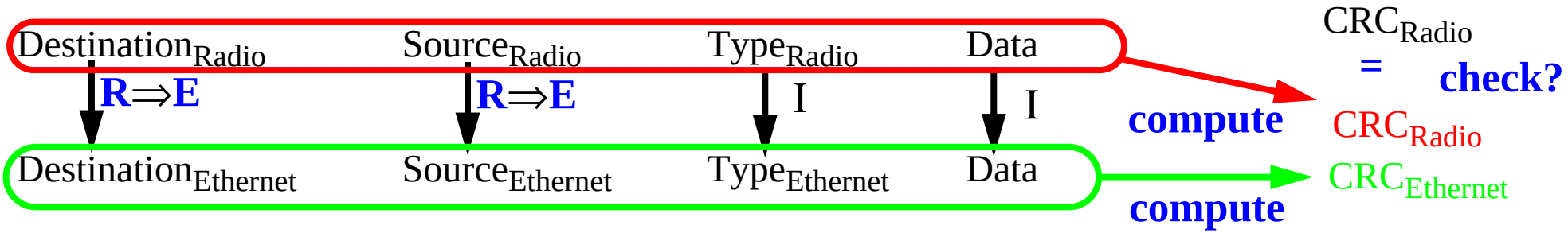


Figure 33: Frame in from Radio out to Ether

The dual of the above holds for frames going from Ethernet to Radio.

Processing:

- upon receiving a **frame** we check the arriving CRC if it is wrong, the frame can be immediately dropped
  - ♦ note that this is not essential -- as we assume that the frame contains a packet from a higher layer protocol, therefore if the higher layer wants reliability it must in any case check for itself
  - ♦ But avoiding dropping packets **unless necessary** is very desirable to avoid bad effects at other layers
- otherwise replace the source and destination addresses by the address approriate to this media
- compute new CRC
- emit frame

Two forms of the bridge

Note that the transformation of R⇒E can have one of two forms:

- transparent (i.e., an identity transformation) or
- non-transparent (i.e., turning one sort of address into another)

Thus in the transparent form, MAC address will be unchanged.

Bridging Data Structures

Figure 34: Bridging Data Structures

Ethernet MAC address	Radio MAC address
0800090056f0	08000900a953
08000900565a	08000900161e
ffffffffffff	ffffffffffff

Bridging processing

Figure 35: Sample non-transparent Bridging MAC-MAC table

Ethernet MAC address	Radio MAC address
0800090056f0	08000900a953
08000900565a	08000900161e
ffffffffffff	ffffffffffff

X to Y (where $X, Y \in \{Ethernet, Radio\}$):

- **if** $Packet_{destination\ address}$ = X MAC address[i] in the table
- **then** replace $Packet_{destination\ address}$ by Y MAC address[i],
  - **if** $Packet_{source\ address}$ = X MAC address[j] in the table, **then** replace $Packet_{source\ address}$ by Y MAC address[j],
    else ignore Packet
  - compute new CRC, emit Packet
- **else** ignore Packet

If we assume that we have a pool of Ethernet MAC addresses for use at this access

point, we assign a new Ethernet MAC for each new Radio MAC address which we are going to serve. [Note that if the Radio MAC addresses are the same lengh as Ethernet MAC addresses (and non-overlapping spaces) - we could just use the same MAC address on each side and hence operate as a transparent bridge).

Note that if the destination MAC address is not in our destination table (but is known to be on the same side as the source, then we just ignore the packet) otherwise we have to decide where to send it. However, in the case of a two port device it is either on one side or the other, so if this address has not been seen as a source on the same side as the current source address, we can just forward it out the other port!

Important Operations for Bridging

As can be seen from the algorithm the operations done on each frame are:

- field extract
- associative lookup of Y given X
- field replacement
- CRC computation
- Filtering

Since the fields are in fixed positions - it is trivial to do an extract or replacement.

Why the “ffffffffffff” entry in the table? This is the link broadcast address, it maps to the broadcast address on the other link. It is used by the higher layer processes to get a specific link layer address. In the case of IP, this is done via the ARP protocol.

Note that new devices which are to be served by an access point initially use the broadcast MAC address to get service - as they don’t necessarily know any other link address besides their own.

Filtering

Proxim's RangeLAN2 7510/7520 Ethernet AP-II Access Point supports the following filters:

- Fixed Nodes Filter (i.e., eliminate specific nodes)
- Protocol Filters
  - IP, IPX, NetBEUI, DECNET, AppleTalk, Other ¿?
- Broadcast Traffic Filters
  - IP/ARP: filter needs Network IP address and Netmask
  - Novell RIP, SAP and LSP
  - Adjustable Bandwidth Allocation

How many link layer addresses are necessary?

Multiple link address

- N+1 links: 1 per remote (i.e., client) + 1 for the device itself
  - demux based on the link address - thus we know which client to send which frame to - directly from the link layer address
  - this requires that we havea pool of link layer addresses which we assign to the clients OR that we proxy with the clients link address (since each client already has one link address)
  - the 1 address for the device itself allows us to manage it
- 2 links : 1 for all remotes (i.e., clients) + 1 for the device itself
  - we simply output onto the other interface all packets
  - the remotes have to sort things out for themselves (perhaps using high layer protocols)

Note that the difference is the basic difference between **link** layer addressing and **network** layer addressing.

If you have only one link address?

Then either:

1 no management of the device itself (it simply forwards everything that is sent to this address)

2 you must have another packet type or link service access point (LSAP), which you can demultiplex based on, to talk to the device itself

- Note that IP does not know about these other LSAP values, it always uses the value 0x0800 in the type field
- Hence you can't route these other frame types across an IP network, so you will not be able to do remote management! (unless you bridge) Therefore, MR would have to be on the same segment - but this **may** scale poorly, since we would have to have an MR on every segment {unless we can find a way to make the MR's cheap - or even partition them - but can now use an IP protocol from the MR to elsewhere}
- Alternatively one could tunnel these frames inside IP packets from a remote management station, then decapsulate at the MR. This would still require that you have to have a way to demux them when they get to the access point.

3 Use the sender's link address, which feeds a local IP layer

- Then we could do the demultiplexing at the network layer ⇒ possibly more processing

4 Much better to assign the access point device a link address

- Then we can have a separate Ethernet-to-Ethernet state machine and thus separate resources - since none of these frames would ever goto the other interface.
- The result is 3 separate machines - which share some link (i.e., MAC) processing - but are otherwise independent.

MACs need not be complex

The Ethernet 1.0 spec. is two pages long - including the specification of the physical connectors, cable specs, etc.

Discussions with various semiconductor manufacturers has yielded:

- Most of the complexity is to deal with corner cases
- It can all be done digitally - this eliminates the need for an external (analog) transceiver chip
  - 3Com macro cell for ethernets
  - Lucent seems to be offering its LAN parts as macro cells calling it “Silicon Suite”: *http://www.lucent.com/micro/NEWS/PRESS96/093096c.html*
- A number of manufacturers are internally time multiplexing a processor to support several MAC interfaces - without needing to dedicate a processor to each.
- Rough complexity of a complete ethernet MAC chip is 15-17K gates (based on the listed size from several vendors offering this as Intellectual Property - this is often abbreviated as “IP”)

Interface to Ethernet physical layer

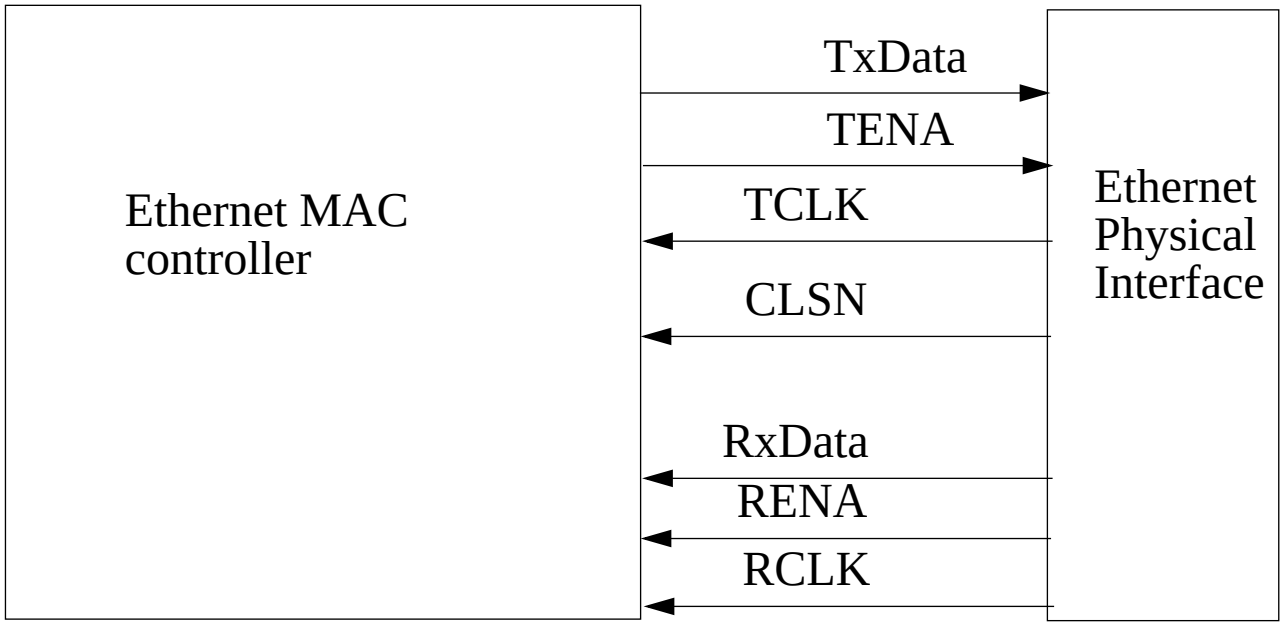


Figure 36: Interface to Serial Interface Adaptor

Tx Data, TENA (Transmitter Enable) {can be seen as Request To Send} are the only outputs, data is sent out when CLSN (Collision) is not busy and is clocked out using Transmitter Clock (TCLK).

RxData, RENA (Receive Enable) - received data is accompanied by a receive clock (RCLK).

Ethernet physical layer encoding/decoding

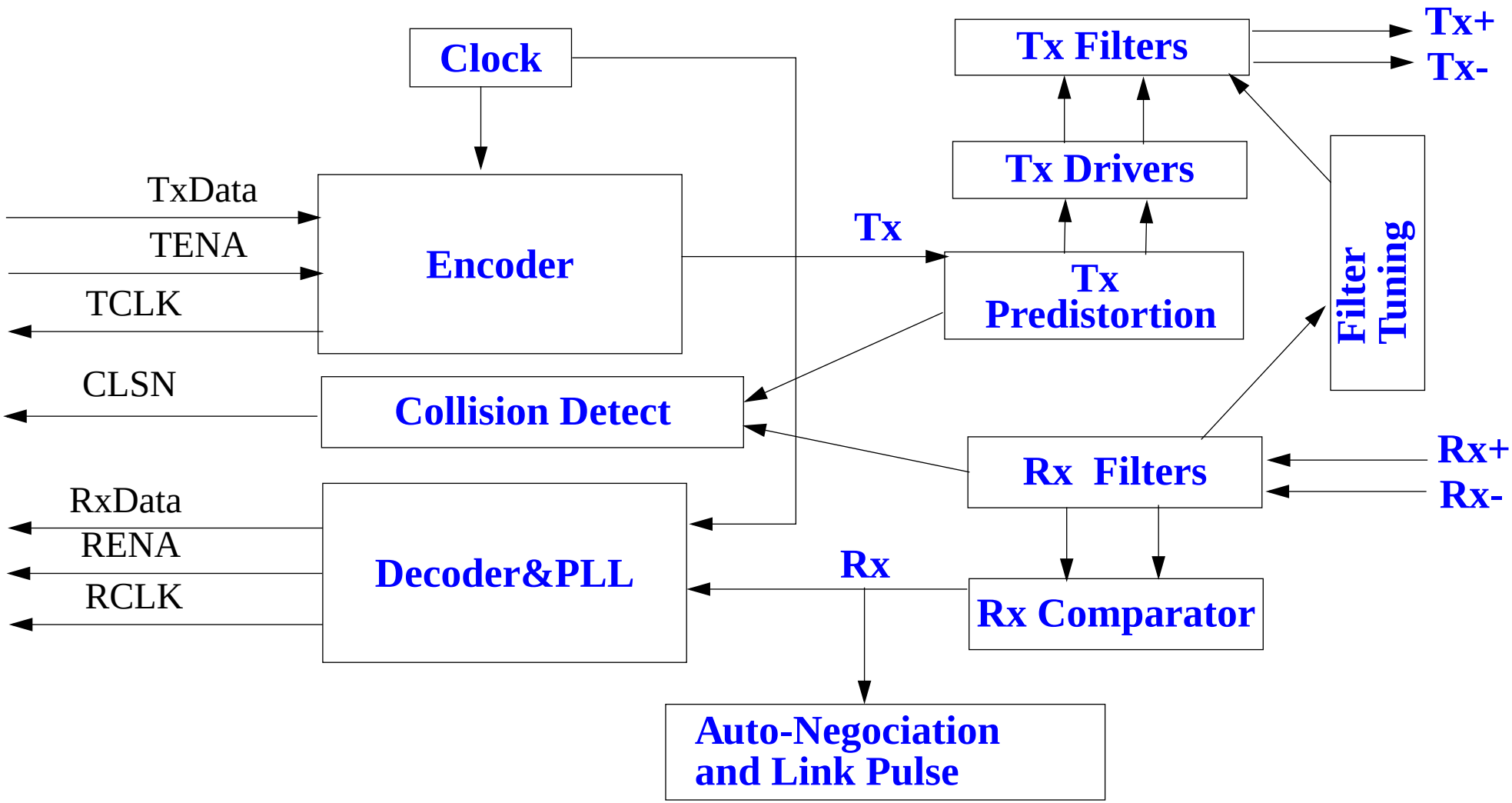


Figure 37: Ethernet phyiscal layer encoding and decoding

Auto Negociation

Use Fast Link Pulses (FLP) according to Ethernet standard (ISO/IEC 8802-3:1995(u)) to transmit a Link Code Word which identifies the capability of the remote device.

Auto-Negotiation can provide automatic speed matching for multi-speed devices (thus a 10/100 Tx interface can detect that the other device is capable of running at 100Mbits/s and choose this speed of operation).

Link integrity test pulses, referred to as Normal Link Pulses (NLP) are periodically transmitted to verify the integrity of the wired link.

A nice introduction to Auto Negociation is available on-line at:
http://wwwhost.ots.utexas.edu/ethernet/100quickref/ch13qr_1.html

A two port Ethernet device

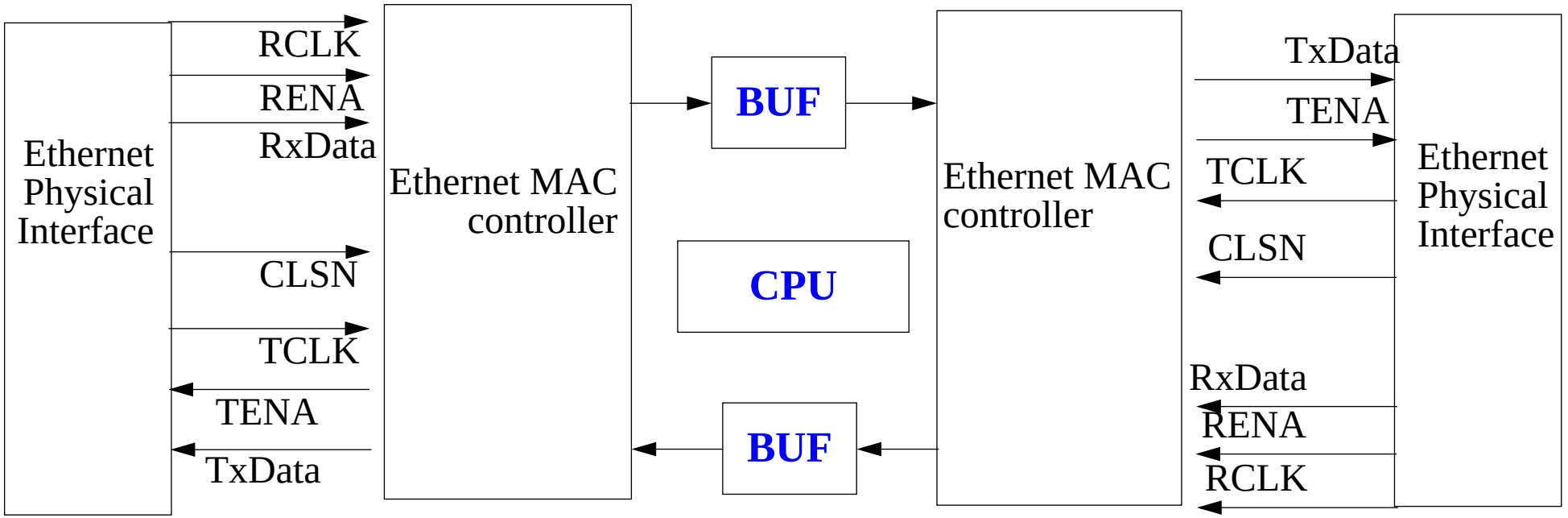


Figure 38: Interface to Serial Interface Adaptor - without local management

To add some control and management we need to add either:

- a second physical interface and Ethernet MAC controller - which can be attached to one of the same physical networks
- a demultiplexer on the output from one of the Ethernet MAC controllers - which recognizes this station's link address

Two port ethernet device with connection for control and management

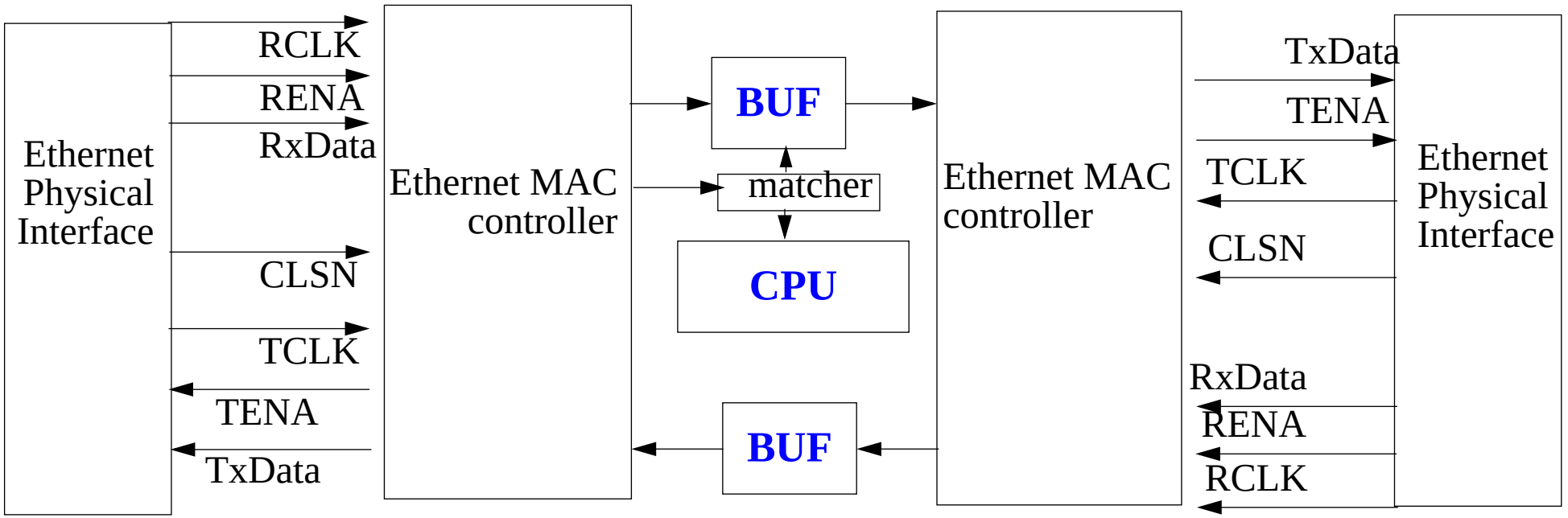


Figure 39: Interface to Serial Interface Adaptor - with control and management

Note that this shows only one network interface being connected for the purpose of control and managment.

Reduced two port device

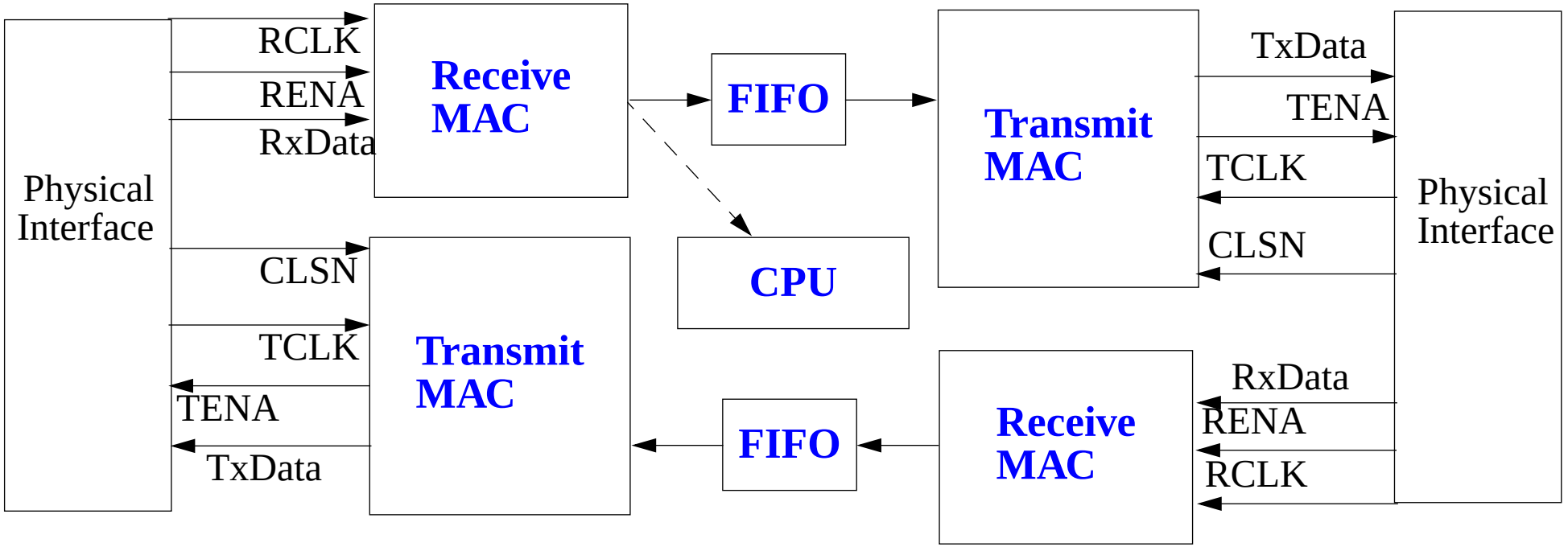


Figure 40: Reduced two port device

The dashed path represents the optional control and management link - either this has some type of filtering on it to detect which frames are destined for this node or the local processor checks this.

Why no need for Ring buffers?

Ring buffers are in the usual ethernet controller:

- to allow the OS to put off taking interrupts all the time, i.e., it can pull a bunch of packets out following one interrupt (assuming that multiple packets have arrived since the last time it looked) and
- to allow the OS to take some time before having to handle the interrupt.

In the OS (for example the BSD TCP/IP code) the packets come out of the ring buffer and are placed in operating system buffers (called mbufs - for message buffers). Then the higher level processing happens. The details for ethernets is in chapter 4 of TCP/IP Illustrated, Volume 2, by Gary R. Wright and W. Richard Stevens.

The ring buffers are actually just a way of implementing the two FIFOs out of one chunk of shared memory (i.e., shared between the ethernet MAC interface and the host computer).

Note that this is NOT true of the Afterburner which HP Labs Bristol built, as in

this case the MBUFs are actually on the network interface card, thus you have to manipulate the pointers and don’t actually have a FIFO (of course you don’t have a ring buffer either! as you just string together the buffers that are ready to transmit and hand a pointer to the lead one to the hardware which follows the pointers outputing the frames as it goes - thus reducing the number of times you touch the data).

Why so simple?

Upon reflection, the reason that our access point can be simple is that packets which arrive are:

- destined for the device itself **or**
- destined for the other interface

There is **no where else** for them to go. If they are destined for the device itself - they can be shoved into a FIFO for this device or fed directly to the local processor - otherwise (**and** this is the usual case) they are put into a FIFO to the other interface which just takes them out of the FIFO and transmits them as soon as the media is free. Since there are no priorities - there is no reason to change the order of the things in the buffer. If there is no room in (either) FIFO, then the frame is simply dropped.

Clearly a major question is how large do the FIFOs have to be, but we also have to keep in mind that simply having longer FIFOs is not a great idea - as it can add to the overall delay of delivering packets and as these FIFOs can be filled to

different depths at different times it great increases the delay variance.

Thus the FIFOs should be large enough to hold the packets that might arrive for each of the destinations which it is currently serving while waiting for the media to be available. The number of packets that might arrive is one window’s worth for each TCP connection and an unknown number of UDP packets.

What if there are three (or more) ports?

Clearly the case of two ports is special as there can be no question of where the frame is to go.

Even extending this to have an “internal” port does not significantly change this picture.

Adding a third port means, either we must

- support more complex buffers (not just FIFOs) **or**
- we have to have multiple FIFOs at each output and then have the output port take frames from the several FIFOs, i.e., it become just like any other switch in needing output buffering

the number of FIFOs at each output port is equal to the number of inputs that could be sending frames to this port - which in the general case is all ports (except perhaps for itself, i.e., number_of_input_ports - 1)

Routing

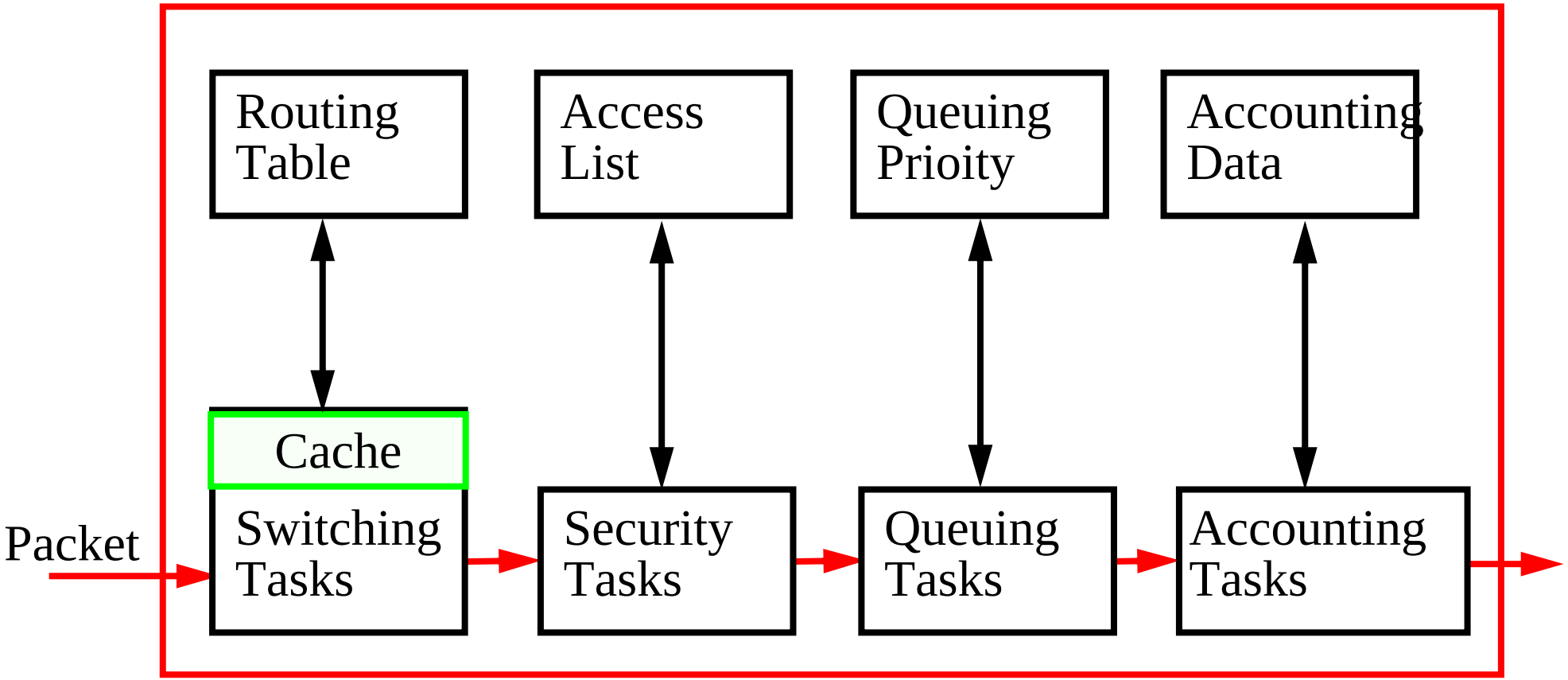


Figure 41: Basic steps in Routing

The routing table tells us which output port to use based on the destination (and possibly the source) IP address.

Routing Principles

- Routing Mechanism: Use the *most specific* route
  - IP provides the mechanism to route packets
- Routing Policy: What routes should be put in the routing table?
  - Use a routing daemon to provide the *routing policy*

For further information see:

Bassam Halabi, *Internet Routing Architectures*, Cisco Press, 1997, ISBN 1-56205-652-2. -- especially useful for IGRP.

Routing table

Flags	Destination IP address	Next-hop Router IP address	point to local interface to use	Refcnt	Use	PMTU …
UGH	140.252.13.65	140.252.13.35	emd0	*ddd*	*ddd*	*ddd*
U	140.252.13.32	140.252.13.34	emd0	*ddd*	*ddd*	*ddd*
UG	default	140.252.13.33	emd0			
UH	127.0.0.1	127.0.0.1	lo0			

where *ddd* is some numeric value.

display the routing table with "netstat -rn"

"r" is for routing table

"n" asks for numeric IP addresses rather than name

Flags:

U	route is Up
G	route is to a gateway
H	route is to a host
D	route was discovered by a redirect
M	route was modified by a redirect

Host vs. router - two behaviors

- Hosts generate or sink IP packets
- Routers forward IP packets

Thus it is possible for a device to be both a host **and** a router.

Unless a host is **explicitly** configured as a router is **not** supposed to forward IP datagrams. The default behavior must be **never forward**.

Host routing

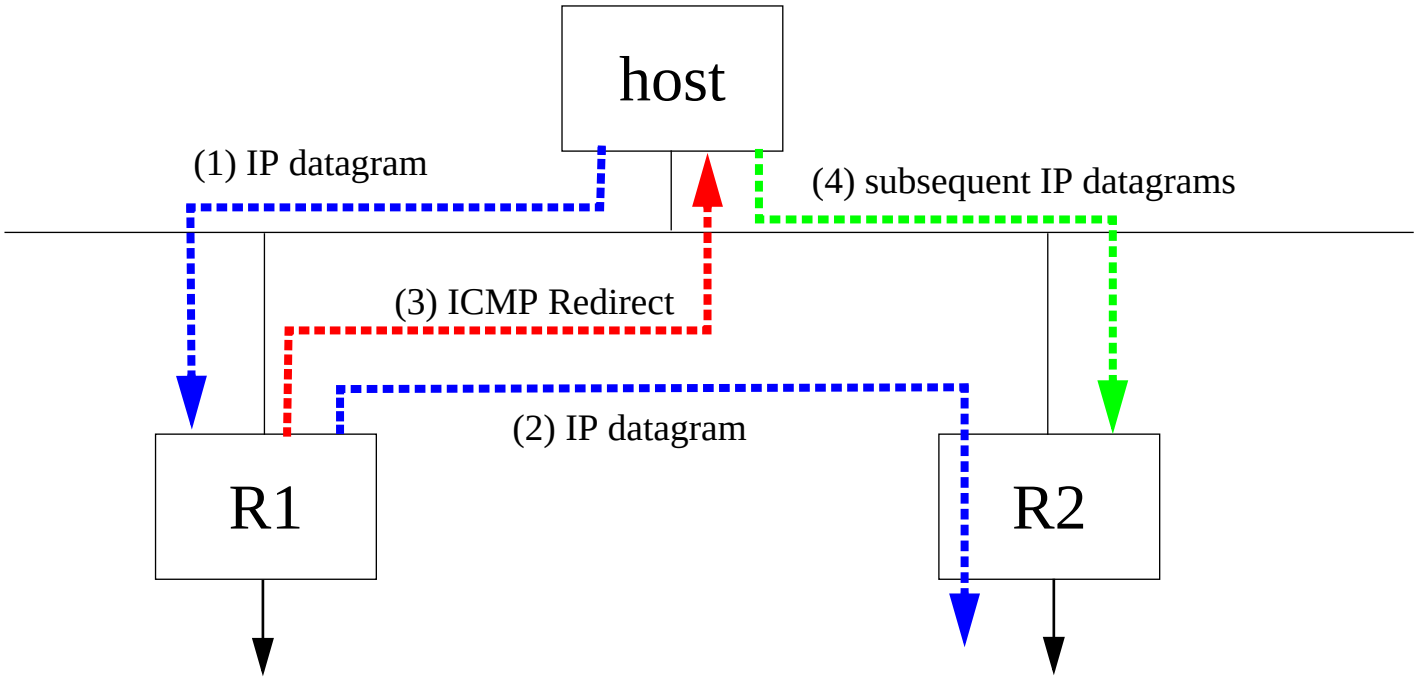
A host either:

- knows a route - manually configured [i.e., "Static routes"]
  - from the interface (for directly connected networks) or manually via the "route" command
- or learns of a route
  - Simplest method of learning a route:
    - The host sends a packet via the default route and is told via an ICMP Redirect of a better route
  - or the host hears an ICMP router advertisement (perhaps in response to its ICMP router solicitation message)
    - routers (almost) periodically broadcast or multicast advertisements of their existence and desire to provide routing service
    - format of ICMP router advertisement packet shown in vol1. figure 9.7, pg. 124
    - advertisements typically every 450..600 seconds
    - advertisements have a stated lifetime (typically 30 minutes)
  - or the host learns via a dynamic routing protocol.
- or uses a default route.

On booting hosts send ~3 ICMP router solicitation messages (~3 seconds apart) to find a default router. This allows for dynamic discovery of the default router.

ICMP Redirect

ICMP Redirect message is sent by a router (R1) to the sender of an IP datagram (host) when the datagram should have been sent to a different router (R2).



Routing packets in the Internet

Router needs to know where to route packets, to do this they need routing information. Such information can be provided by **manually entered routes** or ICMP Redirect or learning of routes via **a routing protocol**.

Dynamic routing protocols are based on routers talking to each other.

- Interior Gateway protocols - within an AS
- Exterior Gateway protocols - between ASs

The most popular dynamic routing protocols are:

- RIP-1 - Routing Information Protocol (version 1)
- RIP-2 - Routing Information Protocol (version 2)
- OSPF - Open Shortest Path First
- BGP - Border Gateway Protocol

Autonomous systems (ASs) - RFC1930

Each of which is generally administered by a single entity.

Each autonomous system selects the routing protocol to be used **within** the AS.

Network	AS number
SUNET.SE	1653
KTH.SE	2836
STANFORD.EDU	32

To find out who is responsible for a given autonomous system, use a query of the form:

http://www.ripe.net/perl/whois?AS8973

For a list of AS number to name mappings (number cache used by the CIDR-Report) see: *http://www.employees.org/~tbates/autnums.html*

Routing Metrics

the measure of which route is better than another:

- Number of hops
- Bandwidth
- Delay
- Cost
- Load
- …

It is possible that the metric uses some **weighted combination** of the above.

Routing Algorithms

- Static vs. Dynamic
- Single path vs. Multi-path
- Flat vs. Hierarchical
- Host-intelligent vs. Router-intelligent
- Intradomain (interior) vs. Interdomain (exterior)
- Link state vs. Distance vector

Interior Gateway Protocols (IGPs)

also called “intradomain routing protocols”

Examples:

- HELLO - an old IGP protocol
- RIP - widely used
- OSPF - increasingly used

Routers that speak any dynamic routing protocol **must** speak RIP and OSPF.

Routing Information Protocol (RIP)

RIP version 1

RFC 1058 {Note it was written years after the protocol was in wide use!}

RIP is a distance-vector protocol - RIP messages contain a vector of hop counts.

RIP messages are carried via UDP datagrams

Message contains {see figure 10.3, pg. 130 of Vol. 1}:

- a command: request or reply
- a version number (in this case 1)
- up to 25 instances of:
  - address family (2 = IP addresses)
  - IP address
  - metric [hop count]

RIP v1 operation

As carried out by UNIX daemon “routed” using UDP port 520

Initialization:

```
for all interface which are up
{
  send a request packet out each interface asking for the other
  router’s complete routing table
          [command=1, address family=0, metric=16
}
```

Request received:

```
if whole table requested, then send it all 25 at a time
else if a specific set of routes
     then fill in the metric
     else set metric to 16
         [16 == “infinity” == we don’t know a route to this address]
```

Response received:

```
if valid (i.e., not 16),then update/add/delete/modify routing table
```

When are routes sent?

If a metric for a route changes, then (trigger) send update,
else send all or part of the table every 30 seconds.

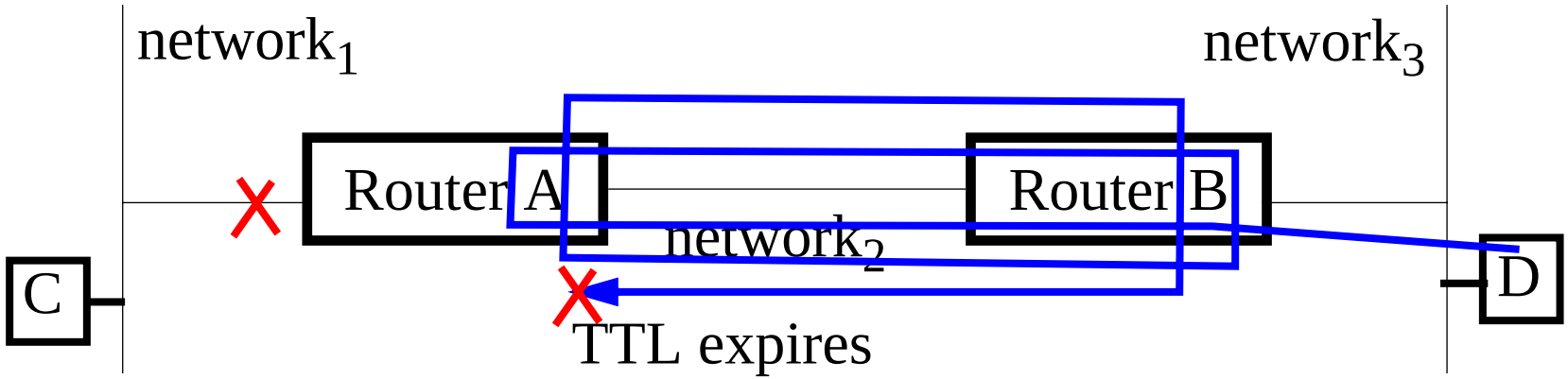
If a route has not been updated for 3 minutes (i.e., 6 update cycles),
then set metric to 16 and then **after** 1 minute delete route.

Metrics are in units of hops, thus this protocol leads to selection between routes based on the minimum number of hops.

Problems with RIP v1

- RIPv1 does not know about subnets (or assumes all interfaces on the network have the same netmask)
- after a router or link failure RIP takes minutes to stabilize (since each neighbor only speaks ~every 30 seconds; so the time for the information to propagate several hops is minutes) ==> while it is unstable it is possible to have routing loops, etc.
- Hops count may not be the best indication for which is the best route
- Since the maximum useful metric value is 15, the network diameter must be less than or equal to 15.
- RIP will accept updates from anyone - so one misconfigured device can disrupt the entire network.
- RIP uses more bandwidth than other protocols, since it sends the whole routing table in updates.

Count to Infinity



- Router A advertises it knows about routes to networks 1 and 2
- Router B advertises it knows about routes to networks 2 and 3
- After one update cycles A and B know about all 3 routes.

If A's interface to $Network_1$ goes down, then A learns from B - that B knows a route to $Network_1$; so A now thinks it can reach $Network_1$ via B. So if D sends a packet for C, it will simply loop back and forth between routers A and B, until the TTL counts down to 0.

Split Horizon

To counter the count to infinity, the split horizon algorithm - never sends information on an interface that it learned from this interface.

RIP version 1 implements: Split Horizon with Poison Reverse Update - rather that not advertise routes to the source, we advertise them with a metric of 16 (i.e., unreachable) - hence the source simply ignores them.

Unfortunately split horizon only prevents loops between adjacent routers

Triggered updates and Hold-Downs

To decrease convergence time - when the topology changes send out an update immediately.

However, if a node can learn about connectivity from more than one source, then if a delete happens before the add, then the existence of the route is asserted; therefore the hold-down rules says:

> When a route is removed, no update of this route is accepted for some period of time - to give everyone a chance to remove the route.
>
> This period of time is the Hold-down time.

The result is to decrease the rate of convergence; but the combined effect of triggered updates + hold-downs leads to **faster** convergence than not using triggered updates.

RIP extensions (aka RIP-2)

Defined in RFC 1388

Version number field is 2.

Added fields:

- routing domain - processor unique ID, so that multiple RIPs can run on one computer - one for each routing domain
- for each of up to 25 entries we add the fields:
  - route tag - carries the AS number
  - subnetmask - subnetmask to be used with this address
  - next-hop IP address, either the IP address of where packets to this destination should be sent or zero [which means send them to the system which sent the RIP message]

If the address family value is 0xffff and the route tag is 2, then the remaining 16 bytes of this entry is a clear text password to be used to authenticate this message.

RIP-2 supports multicast, rather than just broadcasting [to reduce load on hosts - that are NOT interested in RIP-2 messages].

Why would anyone use RIP?

After all these problems you might ask this question.

Answer

Because RIP is generally the only routing protocol which all UNIX machines understand!

Interior Gateway Routing Protocol (IGRP)

Cisco proprietary protocol with the following goals:

- stable, optimal routing for large networks - with no routing loops
- fast response to changes in net topology
- low overhead in both bandwidth and processor utilization
- ability to split traffic across several parallel routes if they are (or nearly are) equal.

It is a distance-vector protocol based on many of the ideas from RIP.

IGRP Metrics

- a vector of metrics each with a 24 bit value
- composite metric is $\left[\left(\frac{K_1}{B}\right)+\left(\frac{K_2}{D}\right)\right]\cdot R$, where K_1 and K_2 are constants, B the unloaded path bandwidth, D a topological delay, and R is reliability
- also we pass the hop count and Maximum Transmission Unit values

K_1 is the weight assigned to bandwidth (by default 10,000,000)

K_2 is the weight assigned to delay (by default 100,000)

If up to 4 paths are with in a defined variance of each other, Cisco's IOS (Internetwork Operating System) will split the traffic across them in inverse proportion to their metric.

IGRP Route Poisoning

IGRP poisons routes which increase by a factor of 10% or more after an update [they are thought to be: “too good to be true”].

While this rule may temporarily delete valid routes (which will get reinstated after the next regular update) - it allows use of a zero hold-down time, which leads to faster convergence.

IGRP Default Gateway

Rather than using the fake network 0.0.0.0 to indicate the default network, IGRP allows a real network to be flagged as the default network.

Periodically, IGRP scans the routes offering a path to this flagged network and selects the path with the lowest metric.

Note: Default gateways help to keep the size of local routing tables smaller.

Enhanced IGRP (EGRP)

Uses the distance-vector technology of IGRP, but changes the way routes are advertised and the calculation of entries for the routing table.

EGRP uses:

- a neighbor discover/recovery process of hello packets to learn its neighbors
- a Reliable Transport Protocol to ensure guaranteed, ordered delivery of routing updates
- a Diffusing Update Algorithm (DUAL) - which selects both the best route for insertion into the table **and** a feasible successor (for if the primary route fails)
- Variable length subnet masks (VLSM)

EGRP is a Cisco proprietary technology.

Open Shortest Path First (OSPF)

OSPF version 2 defined in RFC 1247

OSPF is a link-state protocol. The OSPF messages tell the **status of links** of each of its neighbors and propagates this info to its neighbors. Each router uses the link-state information to build a **complete** routing table.

OSPF uses IP directly (protocol field = OSPF)-hence it does **not** use UDP or TCP.

Advantages

- link-state protocols converge faster than distance-vector protocols
- can calculate a route per IP service type (i.e., TOS)
- each interface can have a per TOS cost
- if there are several equally good routes ==> can do load balancing
- supports variable length subnet masks
- enable point to point links to be unnumbered (i.e., don't need an IP address)
- uses clear text passwords
- uses multicasting

OSPF uses the Shortest Path First algorithm (also know as the Dijkstra algorithm).

OSPF networks generally divided into areas such that cross-area communication is minimal.

Some routers with multiple interfaces become border area routers (with one interface in one area and another interface in another area).

The only way to get from one area to another area is via the backbone - which is area 0. Note: The backbone need not be continuous.

Link state advertisements are sent to all routers in a given area, not just neighbors (as in the distance-vector approach).

A key feature of OSPF is route aggregation - which minimizes the size of routing tables and the size of the topological database; in addition, it keeps protocol traffic to a minimum.

Exterior Gateway Protocols (EGPs)

also called a “interdomain routing protocol” - used between ASs

Examples:

- EGP - an old EGP protocol
- BGP - Border Gateway Protocol

Exterior Gateway Protocol (EGP)

an exterior gateway protocol with three components:

- neighbor acquisition
- neighbor reachibility, and
- routing information

EGP was designed to provide more automation in configuring routers.

EGP is similar to the distance-vector protocols, but omits the metrics, since EGP was designed for the internet where typically routers are connected to a backbone (with its own routing domain) via a single router.

But since there are no metrics, if there is more than one path, then there can be a loop!

BGP - Border Gateway Protocol

An exterior gateway protocol to exchange routing information between routers in different ASs.

BGP version 3 defined in RFC 1267, while version 4 defined in RFC1467

BGP routers exchange routing information with other BGP routers.

For further information see:

JohnW. Stewart III, *BGP4: Inter-Domain Routing in the Internet*, Addison-Wesley, 1999, ISBN: 0-201-37951-1.

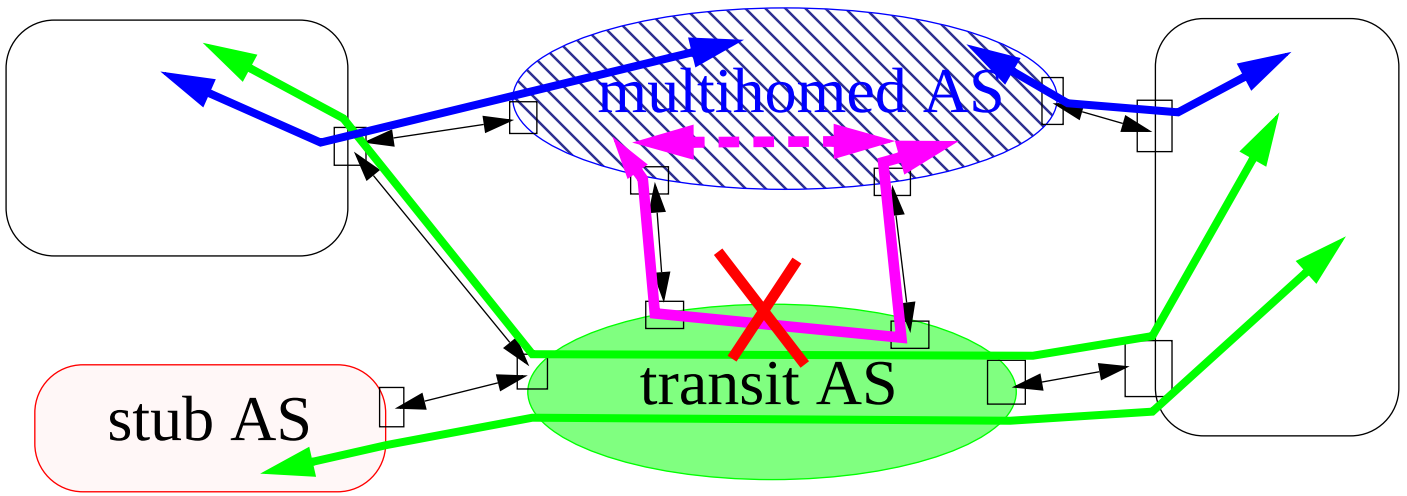
Local vs. Transit traffic

Local traffic: originates or terminates in an AS

Transit traffic: is all other traffic

==> 3 types of ASs

stub AS	connected to only on other AS, carries only local traffic
multihomed AS	connected to multiple AS, but refuses transit traffic
transit AS	connected to multiple ASs, carries both local and transit traffic



BGP operation

BGP routers exchange information based on traffic which transits the AS. Derive graph of AS connectivity; with loop pruning.

Routing policy decisions can be enforced as to what is allowed to transit whom ==> **policy-based routing**

- based on economic/security/political/… considerations.

BGP does **not** implement the policy decisions, but allows the information on which such decisions can be made to propagate as necessary.

BGP uses TCP as its transport protocol

- initially two systems exchange their entire BGP routing table,
- then they simply send updates as necessary.

BGP is a distance-vector protocol - which **enumerates** the route to each destination (i.e., the sequence of AS numbers which a packet would have to pass through from a source to its destination) = a path vector

BGP does not transmit metrics.

BGP detects failures (either links or hosts) by sending **keepalive** messages to its neighbors. Generally sent every 30 seconds and as they are only 19 bytes each ==> only ~5 bits/second of bandwidth.

A major feature of BGP version 4 is its ability to do aggregation - to handle CIDR and supernetting. For more information on aggregation see chapter 5 of “Internet Routing Architectures” by Bassam Halabi, Cisco Press, ISBN 1-56205-652-2.

Classless Inter-Domain Routing (CIDR)

A standard for both classless addressing and classless interdomain routing scheme (RFCs 1517 .. 1520).

- Basic concept: to allocate/collapse a block of contiguous IP addresses into a single routing table entry: (network address, count). e.g., 192.5.48.0, 192.5.49.0, 192.5.50.0 = (192.5.48.0, 3)
- Hierarchical Routing Aggregation minimizes routing table entries; enables "route aggregation" in which a single high-level route entry can represent many lower-level routes in the global routing tables.
  - Reduces the growth of routing table.
- Allows the addresses assigned to a single organization to span multiple classed prefixes.
- Envisioned a hierarchical Internet.

CIDR addressing scheme and route aggregation has two major user impacts:

- you have to justifying IP Address Assignments
- get address from your ISP, i.e., renting them vs. being assigned them

Redistribution of Route Information between protocols

redistribution: allows a router running more than one routing protocol to distribute information from one protocol to another.

Thus at the border, a router will translate the information obtained from one routing domain and pass it to the other routing domain in the appropriate manner.

Interconnections of networks

Since different networks have different users, policies, cost structure, etc.

But, the value of a network is proportional to the (number users)^2 [Metcalf's Law]

Therefore, network operators **want** to connect their networks to other networks.

The first of these were:

- U. S. Federal Internet eXchange (FIX) and
- Commercial Internet eXchange (CIX)

Federal Internet eXchange (FIX)

A top-level routing domain - i.e., it does not use default routes.

Each was built around an FDDI ring which interconnected routers from each of the operators.

Each of these routers was in turn connected to the rest of the operator's network via a high speed link (often at speeds up to 45Mbps).

Note that it need not be a physical ring, but is often an FDDI switch (such as the DEC Gigaswitch/FDDI).

Commercial Internet eXchange (CIX)

A nonprofit trade association of Public Data Internetwork Service Providers.

- a neutral forum - for forming consensus on legislation and policies
- fundamental agree for all CIX members to interconnect with on another
- no restriction on traffic between member networks
- no “settlements” or traffic charges

Global Internet eXchange (GIX)

Global Internet eXchange (GIX), Guy Almes, Peter Ford, Peter Lothberg.

proposed in June 1992.

Stockholm D-GIX - technical *specification*

Netnod Internet Exchange i Sverige AB (*Netnod*)

SNUS (Swedish Network Users Society)

> "... its goal, from the users perspective, to force the evolution and development of the networks and interconnections between networks, to arrange seminars, to exchange information between the members, and to write agreements with companies."

- SOF (Swedish Operators Forum)

Network Access Points (NAPs)

At the NAP a high-speed network or switch is used to interconnect a number of routers for the purpose of exchanging traffic.

http://www.merit.edu/merit/archive/nsfnet/post.nsfnet.html

The 4 NSF sponsored NAPs:

- Sprint NAP - in Pennsauken NJ, FDDI LAN supports both shared and dedicated bandwidth with DEC GIGASwitch-based crossbar switching
- PacBell NAP - in San Francisco, hub for NAP connections in five LATAs, and an ATM (OC3/DS3)/FDDI hybrid
- Ameritech Advanced Data Services (AADS) NAP - in Chicago, ATM (OC3/DS3)/FDDI hybrid
- MAE-East (part of WorldCom) - in Washington DC, a bridged FDDI/Ethernet hybrid
- MAE-West (part of WorldCom) - in San Jose CA, a bridged FDDI/Ethernet hybrid

In addition to handling IP packets, NAPs were required to support InterDomain Routing Protocol (IDRP) {the ISO OSI Exterior Gateway Protocol} and route CLNP (ConnectionLess Networking Protocol) packets. Some MAEs/NAPs

Alaska (AMAP)	Northern VA NeutralNAP	Taiwan (TWIX)
Austin exchange	OhioEP	Vermont (VIX)
Baltimore (BNAP)	PACBELL-SF (PB-SF)	Western Australia (WAIX)
Boston (BMAX/BMEP/BMXP)	PACBELL-LA (PB-LA)	Napsindo (NAiix)
Dallas (DFWMAP)	Palo Alto IX (PAIX)	AUSix Sydney (AUSix)
Denver (MAX)	Palo Alto IX 2 (PAIX-2)	MAE-Europe-ATM (MAE-EU)
Houston (CHNAP/CNAP)	PAIX-Virginia (PAIX-VA)	CALREN-2
Indonesia (idIX)	PAIX-Virginia #2 (PAIX-VA #2)	Pittsburg GP (PSC-GP)
Johannesburg (JINX)	PAIX-Seattle (PAIX-SEA)	TEXAS-GP
KenyaIX	PAIX-Seattle #2 (PAIX-SEA2)	Taiwan (TWIX)
MAE-Chicago (MAE-CHI)	Nap Of the Americas (NOTA)	Vermont (VIX)
MAE-Central-ATM (MAE-Central)	Panama IX	Western Australia (WAIX)
MAE-East-ATM (MAE-EAST)	Paris (PARIX)	Napsindo (NAiix)
MAE-Houston (MAE-HOU)	Philidelphia (PHIL-IX)	AUSix Sydney (AUSix)
MAE-LA (MAE-LA)	Phillipines (PHiX)	MAE-Europe-ATM (MAE-EU)
MAE-New York (MAE-NYC)	Pittsburg IX	CALREN-2
MAE-Paris (MAE-PAR)	Salt Lake City (SLC-REP)	Pittsburg GP (PSC-GP)
MAE-Toronto (MAE-T)	San Antonio (SAMAP)	TEXAS-GP
MAE-West-ATM (MAE-WEST)	San Diego (SD-NAP/CAIDAIX)	Taiwan (TWIX)
MXP-Detroit	Sanjaya (idIX-2)	Vermont (VIX)
MetroIX-NewYork (METROIX)	Seattle (SIX)	Western Australia (WAIX)
Nashville (NNAP)	Seattle (SNNAP)	Napsindo (NAiix)
New Mexico (NMNAP)	Singapore (STIX)	AUSix Sydney (AUSix)

NAPs today

Using FDDI switches or ATM switches or direct attachment of routers to high speed links, or increasingly resilient packet ring (Spatial Reuse Protocol (SRP)) technology, e.g., Cisco's Dynamic Packet Transport (DPT) and dedicated fiber connections to/from members.

NAP managers are increasingly concerned about security, reliability, and accounting & statistics.

Various NAP have different policies, methods of dividing costs, fees, co-location of operators equipment at the NAP, etc.

Netnod establishes and operates National Internet Exchange Points in Sweden

- *http://www.netnod.se/*
- Netnod-IX exchange point in Stockholm (earlier known as the Stockholm D-GIX) - see *http://www.netnod.se/existing.htm*

A look at a NAP/MAE

MAE-WEST

BGP router identifier 165.117.1.133, local AS number 2548

115,367 network entries and 530,250 paths using 29,933,498 bytes of memory

88,941 BGP path attribute entries using 4,624,932 bytes of memory

222 BGP rrinfo entries using 5,328 bytes of memory

22,257 BGP AS-PATH entries using 545,000 bytes of memory

2,134 BGP community entries using 114,526 bytes of memory

313 BGP route-map cache entries using 5,008 bytes of memory

Router Arbiter Project

Router Arbiter (RA) -*http://www.ra.net*

- provide a common database of route information (network topology and policies) [Routing Arbiter Database (RADB)]
- promote stability and manageability of networks

Instead of a full mesh connection between providers, all the providers peer with a central router server.

Router server (RS):

- maintains a database of all information operators need to set their routing policy (written in RIPE 181, see RFC 1786)
- does not forward packets or perform any switching function
- a distributed rover runs at each RS and collects information which the central network management system later queries.

Internet Routing Registry (IRR)

- a neutral Routing Registry
- set of routing DBs which includes
  - RIPE Routing Registry (European ISPs)
  - MCI Routing Registry (MCI customers)
  - CA*net Routing Registry (CA*net customers)
  - ANS Routing Registry (ANS customers)
  - JPRR Routing Registry (Japanese ISPs)
  - Routing Arbiter Database (RADB) (all others)
- entries are maintained by each service provider

Internet Performance and Analysis Project (IPMA) IRR Java Interface
http://salamander.merit.edu/ipma/java/IRR.html

Very high-speed Backbone Network Service (vBNS)

vBNS project created to provide a backbone for the US high-performance computing users and their SuperComputer Centers.

- mostly OC12C, but now adding OC48C links (2.4Gbps)
- connections to all NAPs
- provide for multimedia services (provides multicast)
- participate in developing advanced routing technologies
- supports IPv4 and IPv6

vBNS backbone network
http://www.vbns.net/netmaps/logical.html

There is some interest in using vBNS for inter-gigapop interconnections.

Internet2

http://www.internet2.org/

- creating and sustaining a leading edge network capability for the [U.S] national research community,
- directing network development efforts to enable a new generation of applications to fully exploit the capabilities ofbroadband networks, and
- working to rapidly transfer new network services and applications to all levels of educational use and to the broader Internet community, both nationally and internationally.

NORDUnet, (Dutch) SURFnet, & Internet2 interconnect agreement: Apr. 1999

Cisco's NetFlow Switching

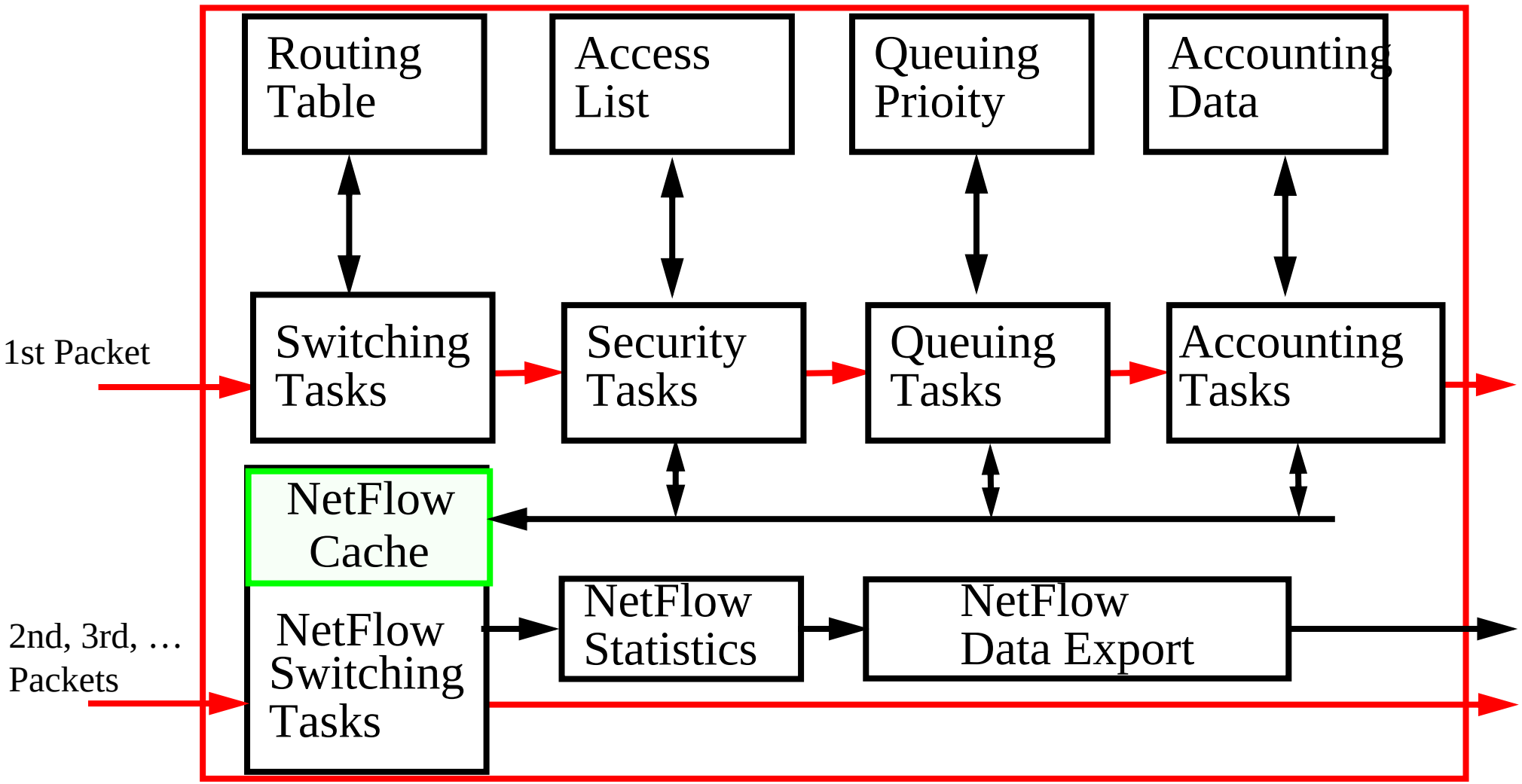


Figure 42: Cisco's NetFlow Switching

Flows

A flow is defined as a “uni-directional stream of packets between a given source network-layer address and port number and a specific destination network-layer address and port number”.

Since many application use well known transport-layer port numbers, it is possible to identify **flows per user per application basis**.

There is a well defined Netflow Switching Developer’s Interface which allows you to get the statistics concerning the NetFlow cache and the per flow data (the later gives you essentially billing records).

A general introduction to NetFlow Switching is available at

http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/switch_c/xcprt3/xcovntfl.htm

Cisco's **Tag Switching**

Combine routing with the performance of switching, based on the concept of "label swapping," in which units of data (e.g., a packet or a cell) carry a short, fixed length label that tells switching nodes how to process the data.

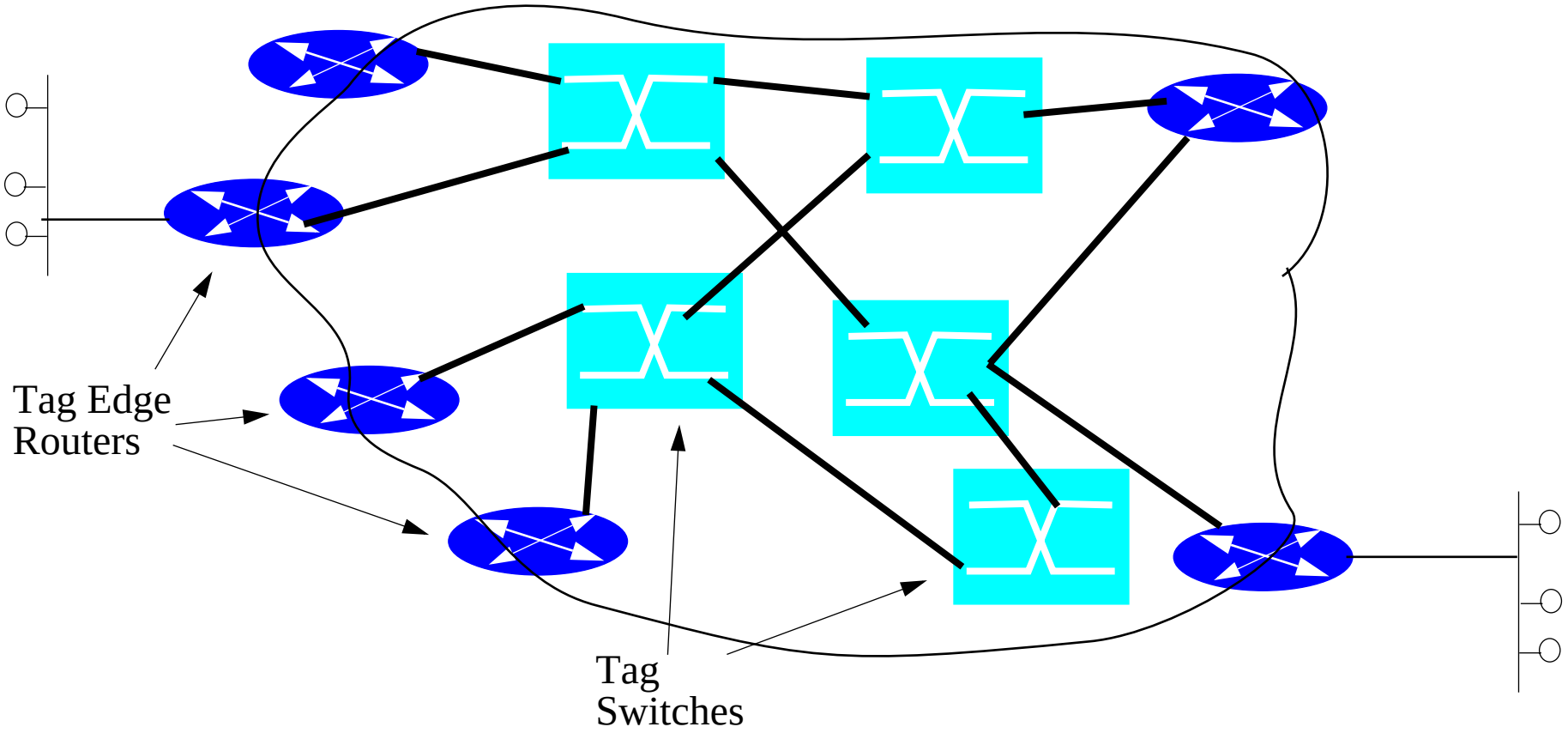


Figure 43: Tag Switching

A Tag Edge router labels a packet based on its destination, then the Tag Switches make their switching decision based on this tag, without having to look at the contents of the packet.

The Tag Edge routers and Tag Switch exchange tag data using Tag Distribution Protocol (TDP).

Basics of Tag switching:

1. Tag edge routers and tag switches use standard routing protocols to identify routes through the internetwork.
2. Using the tables generated by the routing protocols the tag edge routers and switches assign and distribute tag information via the tag distribution protocol (TDP). When the Tag routers receive this TDP information they build a forwarding database.
3. When a tag edge router receives a packet it analyzes the network layer header, performs applicable network layer services, selects a route for the packet from its routing tables, applies a tag, and forwards the packet to the next hop tag switch.
4. The tag switch receives the tagged packet and switches the packet based solely on the tag.
5. The packet reaches the tag edge router at the egress point of the network, the tag is stripped off and the packet delivered as usual.

Tag Locations

- in the Layer 2 header (e.g., in the VCI field for ATM cells)
- in the Layer 3 header (e.g., in the flow label field in IPv6) or
- in between the Layer 2 and Layer 3 headers

Creating tags

Since tag switching decouples the tag distribution mechanisms from the data flows - the tags can be created:

- when the first traffic is seen to a destination or
- in advance - so that even the first packet can immediately be labelled.

Further information on tag switching is available at:

http://www.cisco.com/cpress/cc/td/cpress/fund/ith2nd/it2423.htm

Fast forwarding

Mikael Degermark, Andrej Brodnik, Svante Carlsson, Stephen Pink, “Small Forwarding Tables for Fast Routing Lookups”, in Proceedings of the ACM SIGCOMM’97. (*compressed postscript*) {basis for *Effnet AB*}

- IP routing lookups must find routing entry with longest matching prefix.

Networking community *assumed* it was impossible to do IP routing lookups in software fast enough to support gigabit speeds - but they were wrong!

Paper presents a forwarding table data struct. designed for quick routing lookups.

- Such forwarding tables are small enough to fit in the cache of a conventional general purpose processor.
- The forwarding tables are very small, a large routing table with 40,000 routing entries can be compacted to a forwarding table of 150-160 Kbytes.

- With the table in cache, a 200 MHz Pentium Pro or 333 MHz Alpha 21164 can perform >2 million lookups per second.
  - A lookup typically requires less than 100 instructions on an Alpha, using eight memory references accessing a total of 14 bytes.

∴ Full routing lookup of each IP packet at gigabit speeds without special hardware

Routers do more than routing

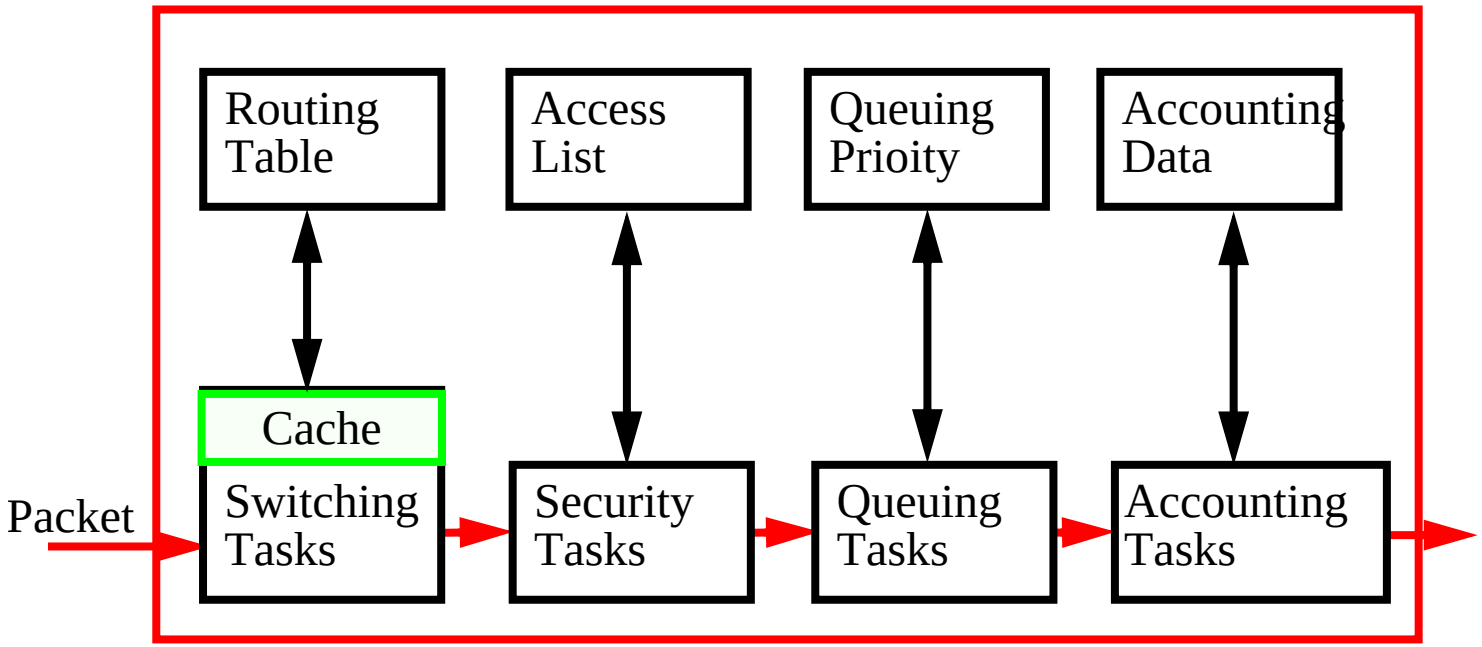


Figure 44: Basic steps in Routing

Earlier we have looked at the routing step, but today much of the excitement is in the details of the other functions. For example, in order to support various QoS features you might want to use more sophisticated queue management: Weighted Round Robin, Fair Queuing, Weighted Fair Queuing, Random Early Detection (RED), Weighted RED, … .

Summary

This lecture we have discussed:

- ARP, RARP
- tools: Ping, Traceroute
- IP routing
  - Routing Principle, Routing Metrics, Routing Algorithms, Routing table
  - Autonomous systems (ASs)
  - Route IP datagrams in a network
  - Dynamic routing protocols
    - RIP, OSPF, BGP, CIDR
  - Cisco's NetFlow Switching and Tag Switching
- NAPs and other interconnect points
- Mentioned some of the high speed networks and interconnect