IP layer - Internet Proto

TCP/IP class

Jim Binkley

IP - Internet Protocol

- intro
- IP addresses
- subnetting
- header
  - fragmentation, ttl, options
- routing/algorithms/architecture
- CIDR/IPng

Jim Binkley

intro - IP version 4

- fundamental TCP/IP protocol
- RFC 791, other related RFCS
  - Inet checksum, rfc 1071, 1141, 162
  - path mtu, rfc 1191
  - ip datagram reassembly 815
  - 1122, communications

ip layer - fundamental idea

- ip implements a **ip virtual netwo**
  of different kinds of hw where ip
  endpoint
- hw is hidden by network layer (ex
  few things like MTU)

what does ip do (and not do?

- sends and recvs packets to/from addresses - **ip datagrams**
- no retries, doesn't promise reliabl
- packets due to various reasons ma *duplicated, delayed, delivered ou or corrupted -*
- **best effort** - don't lose them on p only when nets busy - resources u

ip functions

- route packets
  - routing: process of determining path for
  - ip routes packets when they come to it fr
    - » transport layer (down stack)
    - » link layer (up stack) - we are router a pkts
- fragmentation acc. to link-layer MTU
- handle ip options
- send/recv ICMP error and control mes

ip address

- 32 bits, “dotted-decimal” notatio
  - 1.2.3.4, big-endian byte order, 0..2
- associated with interface, not mac
- if machine > 1 i/f, then **multi-hor**
- if multi-homed, not necessarily ro
- ip address in UNIX assigned to i/
  - #ifconfig ed0 inet 131.253.1.2 netmask

ip address structure

- each address has structure in it: (network, subnet, host)
- classically address consists of (ne portions
- **subnet mask** used to determine s
  - taken from host bits
  - ipaddress & subnet mask

ip address table (net/host)

type	**prefix**	**bytes**	**range**
class A	0	1 net:3 host	1-126.h
class B	10	2:2	128-19
class C	110	3:1	192-22
class D	1110	flat	224..23
class E	11110	-	240..25

class D: multicast

class E: experimental (unused at present), not for broadcast

ip addresses - examples

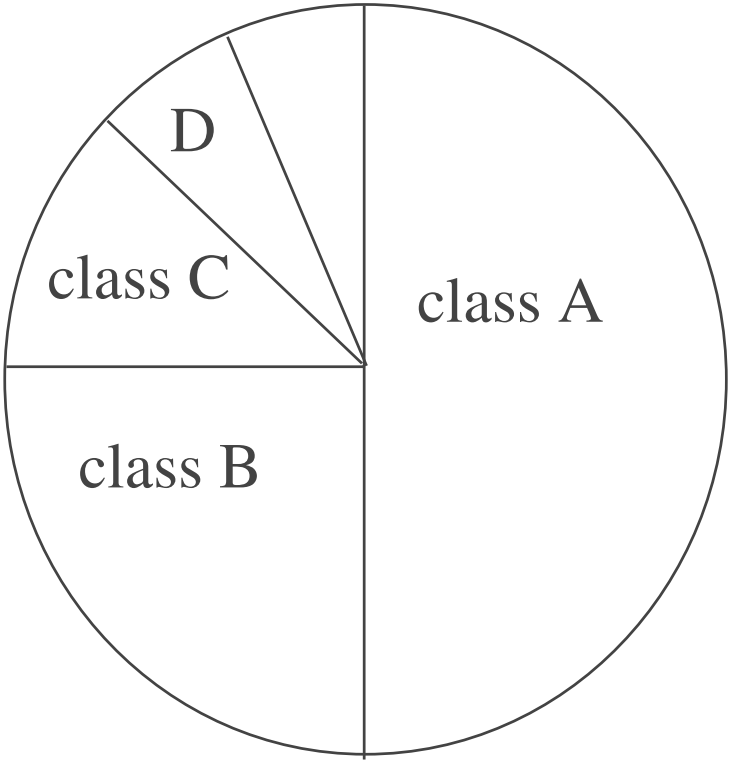
- 0.0.0.0 - if src, then boot == “this net, this ho
  if dest, old 4.2 BSD broadcast addr
- 127.0.0.0 - localhost (loopback)
- 1.2.3.4 - class A
- 143.1.2.3 - class B
- 201.1.2.3 - class C
- 224.0.0.1 - multicast
- 255.255.255.255 - **limited broadcast**
- 200.0.1.255 - **directed broadcast** (assume s
  C part)

ip address - problems

- assigning class by bit means class
  1/2 of range, class B 1/4, class C
- problems with current setup
  - class assignment is wasteful
  - ip host addresses not necessarily uti
  - too many networks in core routers
  - running out of ip addresses ??

the ip pie - not a picnic

ip pie slices
acc. to Class



why was this a mistake?

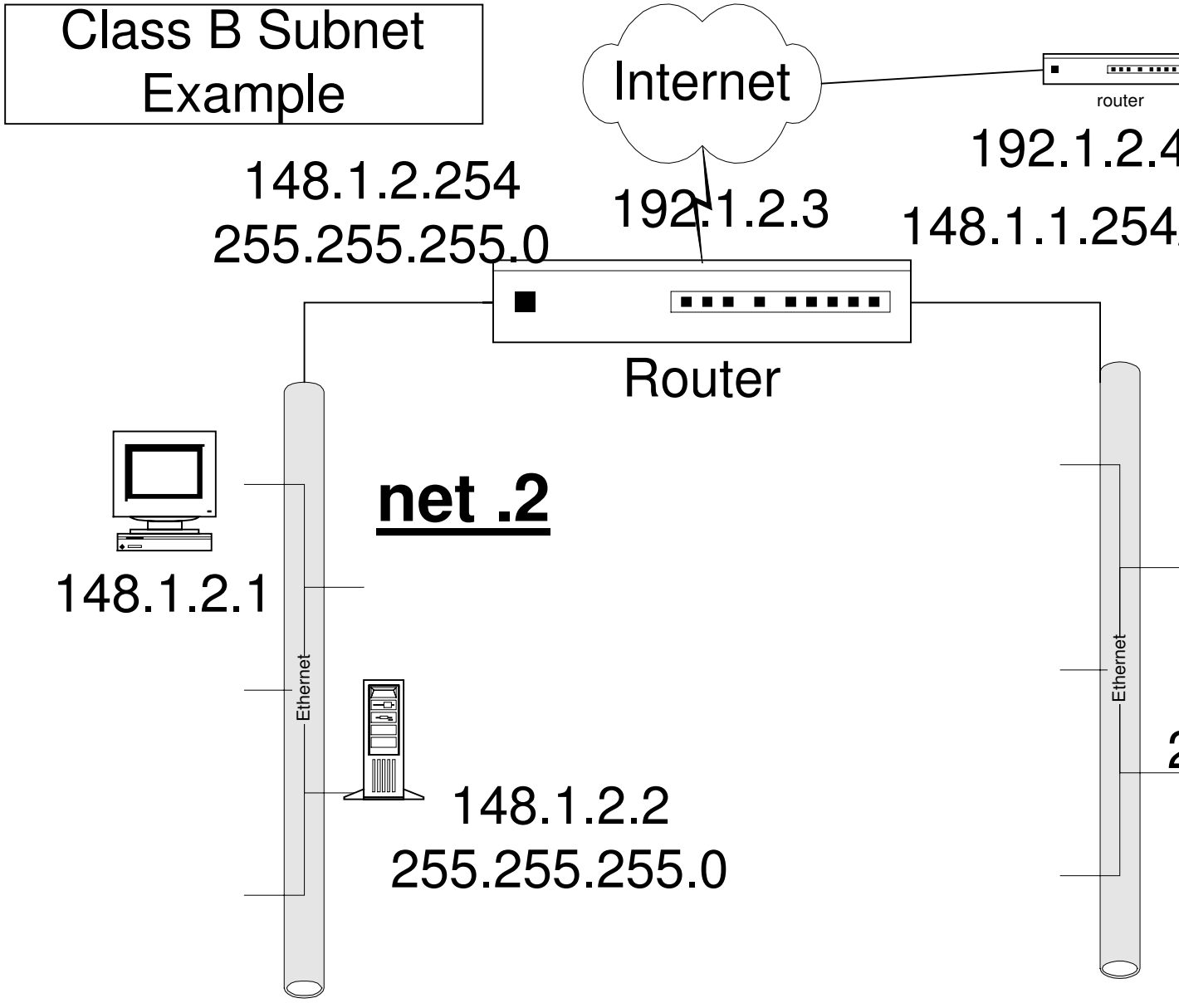
Jim Binkley

subnetting

- subnet - use single IP network ad
  hide multiple physical nets
- subnet notion converts (net, host)
  slightly more hierarchical (net, su
- associate subnet mask with i/f ip
- Example, class B, one byte of sub
  ip = 148.1.1.1 subnet=255.2

subnetting

- subnetting functions:
  1. you can subnet an ip address and sp
  separate networks across routers (cons
  address space)
  2. you hide your routing structure fron
  routers, thus reducing routes in their ro
- *if dest ip addr & subnet mask == my i*
  subnet mask
  dest is on same subnet
  else on different subnet (send pkt to ro

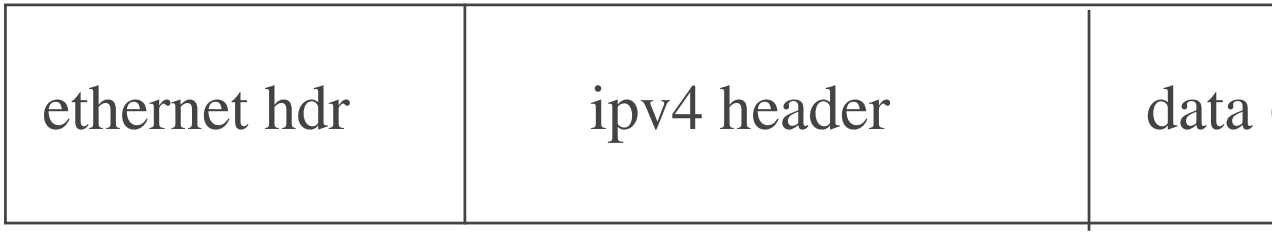

Class B Subnet Example
Internet
router
192.1.2.4
148.1.2.254
255.255.255.0
192.1.2.3
148.1.1.254
Router
net .2
148.1.2.1
Ethernet
148.1.2.2
255.255.255.0
Ethernet

subnetting homework assign

- assume your ISP gives you 1 class C networ 200.0.1.X
- assume your router's ip address to outside w is 134.3.1.2
- design a subnetting scheme that will allow a 30 hosts per subnet
- draw a picture, show ip addresses and subne router ports. Assume the router ports == #s
- show one host on one of the subnets with ip subnet mask, and broadcast address

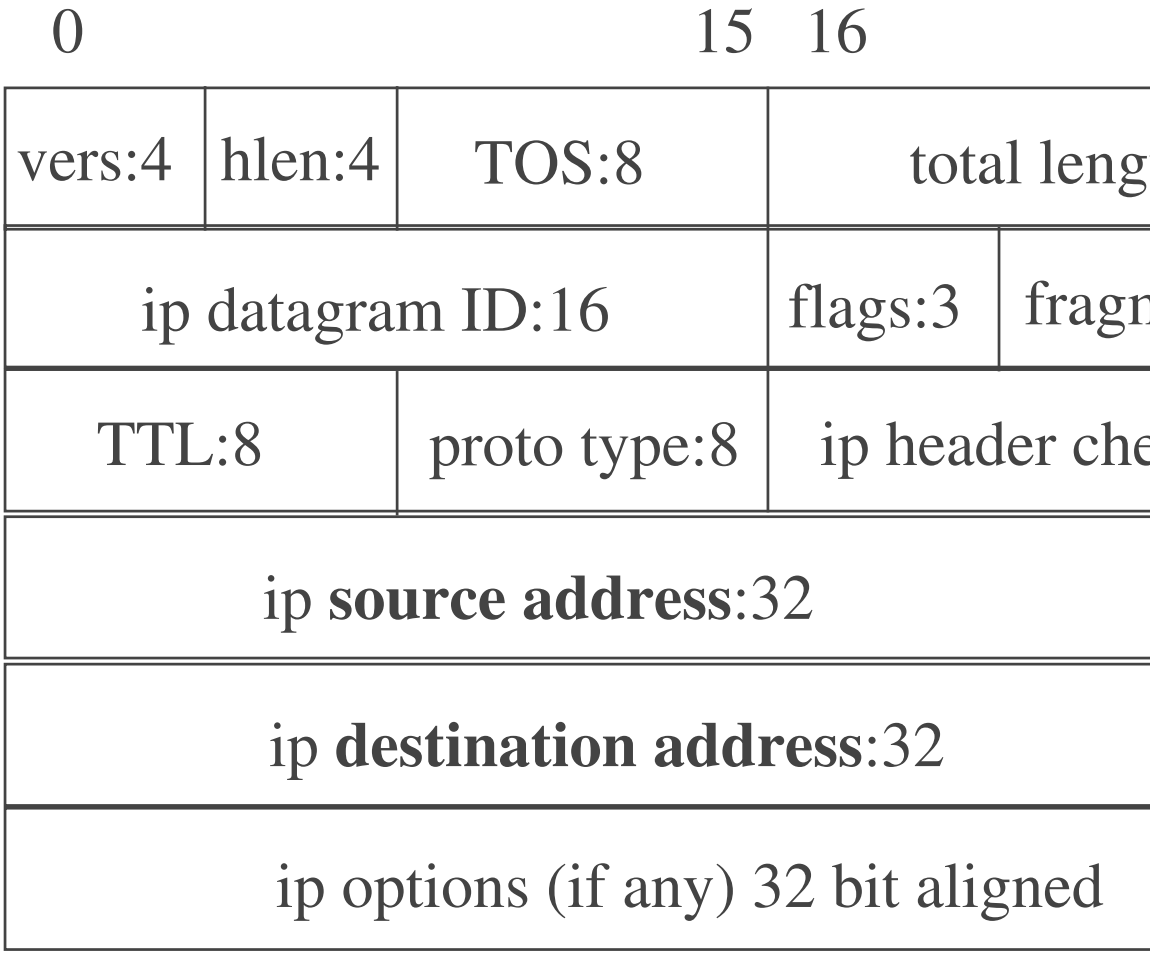
Jim Binkley

ip encapsulation



20 bytes (no options)

ip header


0
15
16
vers:4
hlen:4
TOS:8
total leng
ip datagram ID:16
flags:3
fragn
TTL:8
proto type:8
ip header che
ip **source address**:32
ip **destination address**:32
ip options (if any) 32 bit aligned

ip header

- ip version == 4
- header length in 32-bit words, h == 5 w
  options (20 bytes)
- type of service and precedence
  - not used much in past but starting to be u
  - bits 0-2, precedence
  - bits 3-5, TOS, hint to routing about how t
    - » D (bit 3) - low delay (telnet),
    - » T (4) - high thruput (FTP), R (5) - re

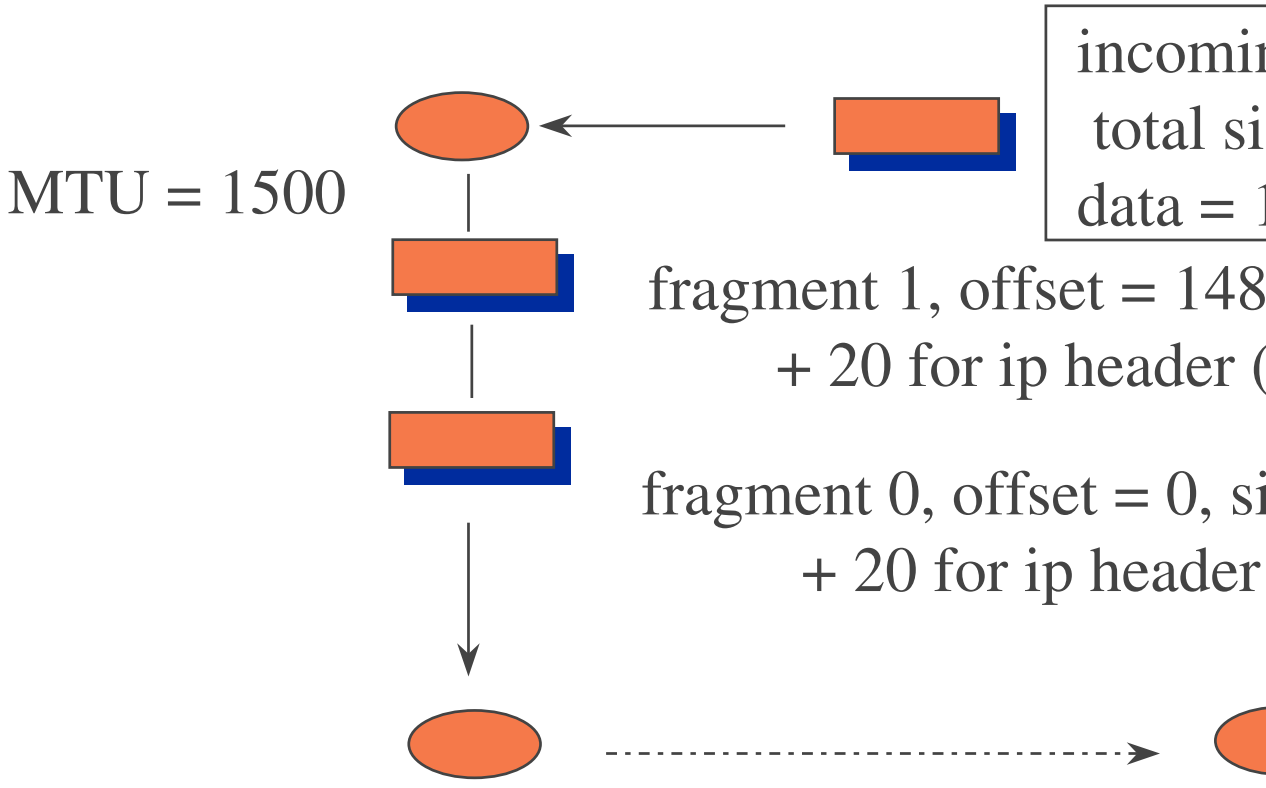
ip header

- total length - max ip datagram is 64k
- fragmentation
  - **fragment ip_id stays the same for all fr**
    - » ip_id ids the logical IP datagram and a
  - flags (DONT_FRAGMENT, MORE_FR
  - fragment offset from 0 start of packet, e.g
  - 0, 0x400, 0x800
  - ip length is length of fragment, not total d

fragmentation - how it works

- ip fragments because outgoing packet
  for MTU of i/f
- fragments must be reassembled at fina
  destination and can be fragmented aga
- if any fragment lost, all of datagram m
  resent (not by IP)
- IP uses best effort even to allocate inte
- TCP tries to avoid, UDP not smart en
- IP fragmentation not a STRONG mecl

ip fragmentation



Jim Binkley

ip_id, ip_src retained in all (new) f

even more fragmentation

- reassembly done at ultimate desti
  - pros:
    - » simplicity - fragments can be routed i
    - » simplicity - intermediate routers don't
  - cons:
    - » any fragment lost, entire datagram los
    - » philosophical - is this really hiding the
- path MTU is a way around
- note: routers may not see all frag

fragments and the MSS

- MSS - **maximum segment size =**
- IP fragmentation up to this size g to work
- UDP apps often do not exceed thi
- TCP often uses PATH MTU, igno size - TCP will try local MTU (of and try to make it work end/end

ip header

- proto type - TCP, UDP, ICMP
- checksum
  - over header only, useful?
  - same algorithm used by tcp/udp
  - with ip itself, only over header
    - » deemed not useful in IPv6
  - routers must redo IP checksum sinc changes

ip checksum

- sender
  - ip checksum field = 0
  - add together by 16-bit words and ta compliment (1's compliment arithn
  - store in ip cksum field
- recv
  - adds N sections, complements resul get 0, else error

ip header - ttl

- TTL - time to live, actually hop c time
- when packet crosses router
  - ttl--
  - if ttl == 0
    - » discard and send ICMP ttl exceeded to
- **important guarantee that datag be discarded even if network lo**

ip options

- not very used and possibly not ve
- variable length encoding mechan
- form of TLV or tag/length/value
- options come in multiples of 32 b
- ignore packet format details
- pro: extensible format
- con: not as easy to parse as fixed

option types

- end of option list
- noop - used for alignment
- military ?
- loose source routing: specify inex
- strict source routing exact path (w addresses)
- record route - possibly useful
- gather timestamps

sassing options

- encoding is not efficient for route
- length is limited by IP header leng
  big enough for size of Inet diamet
  - 60 bytes max, if 4 byte IPs == 15 ip
  - e.g., record route could not work >
- source routing not secure -- some
  stick in a intermediate route a
  your packets

routing

- **routing - the process of choosin over which to send datagrams**
- hosts and routers route
- input: ip destination address
- output: next hop ip address
  and internally an interface to s
- routing does not change ip dest a

how do routes get into routin table?

- static routes - by hand, on unix w
  % route to_dest via_next_hop
- dynamically via routing daemon,
  gated on UNIX, protocols=RIP/(
- via ICMP redirect

show routing table

- unix host
  - % netstat -rn
    - » n is for NO dns, else you may cause D
- cisco router
  - (router) show ip route

routing table

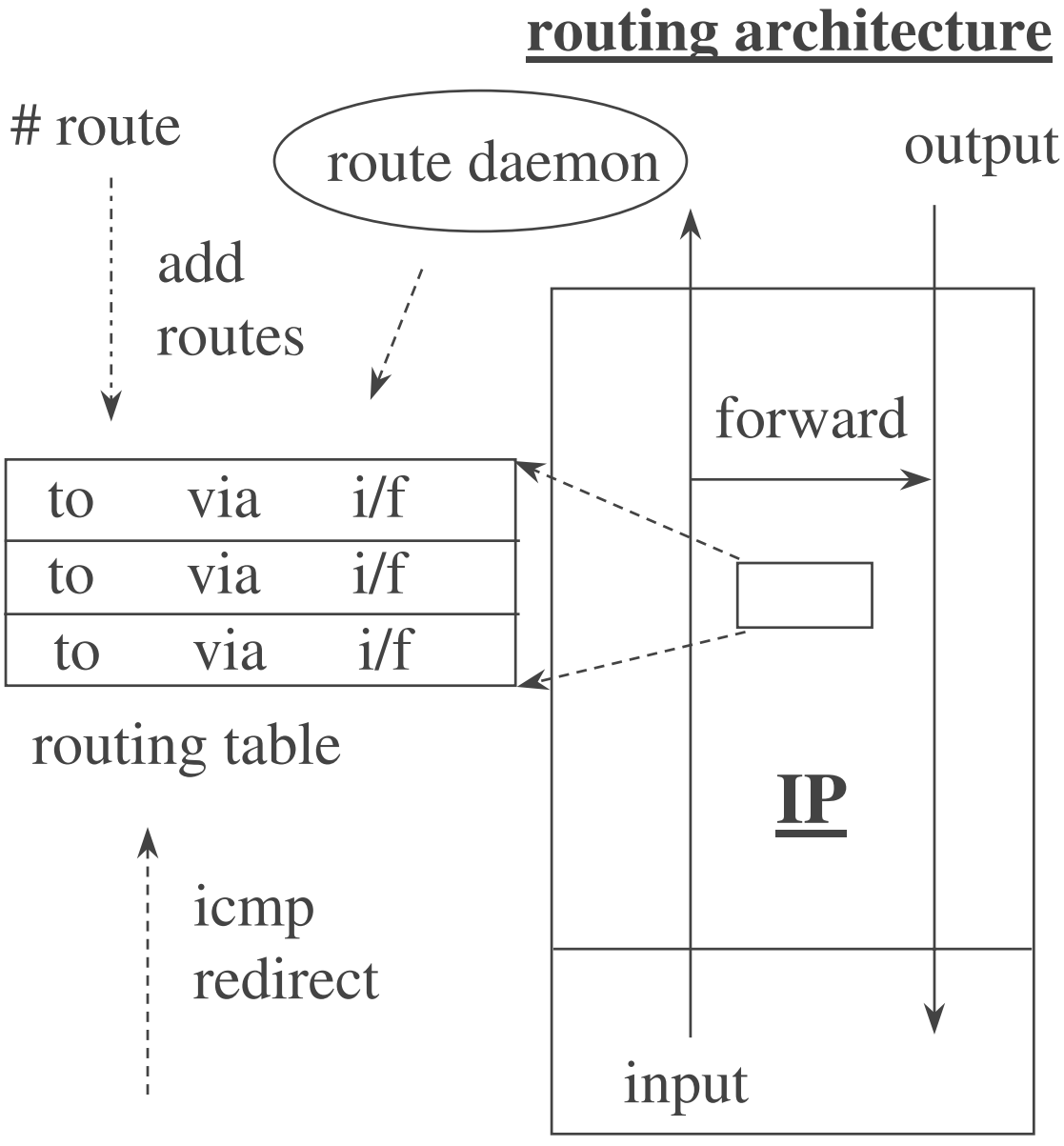
- entries logically
  (destination, mask, via gateway, met
- destination - network or host address
- mask - subnet mask for dst address
- via gateway - next hop (maybe router)
- metric/s - depends on routing table alg
  dynamic routing protocols

manual adds to routing table

- on FreeBSD unix host:
  - # route add default 204.1.2.3 (defau
    - » internally default has value 0.0.0.0 (al
  - # route add 1.1.1.1 2.2.2.2
    - » 2.2.2.2 is the next-hop router for 1.1.1
    - » we must have direct connection to 2.2 be on same subnet and must exist)
    - » # ifconfig ed0 2.2.2.1 (our i/f must e

SOME possible kinds of rou

- host, 210.1.3.21/32 (to specific h
- subnet, 131.253.1.2/24 (to specifi
- network, 131.253.0.0/16 (to spec
- default route - normally the route
  send it here when nothing else ma
  - expressed internally as 0.0.0.0
- note: default route to host route -
  specific to most specific (natural


routing architecture
route
route daemon
output
add
routes
forward
to via i/f
to via i/f
to via i/f
routing table
IP
icmp
redirect
input
transpo
ip jus
table
exter
chan
link lay

netstat -rn - UNIX example

Destination	Gateway	Flags	Rfct
127.0.0.1	127.0.0.1	UH	4
131.252.20.0	131.252.20.183	U	148
131.252.21.0	131.252.21.183	U	92
default	131.252.20.1	UG	1
131.252.10.17	131.252.20.2	UGHD	0
192.220.224.0	131.252.20.1	UG	0
192.147.168.0	131.252.20.1	UG	0
158.104.0.0	131.252.20.1	UG	0
192.147.160.0	131.252.20.1	UG	0

note: and more, RIP in use

routing algorithm/s

- there is no *one* routing algorith
- over time, notion that match shou longest prefix has come into bein
- default < net < subnet < host < br
  - host is longer prefix than net than d
- algorithms vary from (too) simple hardware-assisted (commercial ro

some examples

- dumb pc routing algorithm (exam
- linux (1.2.x/1.3.x) - linear search
- old BSD style, 2 tables, host > ne routes to i/fs in table, hashing for
- new BSD style, Patricia tree algo supports longest matching prefix
- commercial routers - proprietary, support

dumb pc algorithm

assumptions:

consider “to” part, the KEY for every entry

Assume one i/f and we have its subnet mas

we have one default router ip address

algorithm:

if (ip dst & subnet mask) == (KEY ip & su

do “direct” routing; i.e., deliver to i

else

do “indirect” routing, send to defau

Note: this algorithm implies you can’t talk to a d
on the same link

Jim Binkley

supernetting/CIDR

- early 90's -- too many class B ad
  given out - running out of ip netw
  addresses?
- decision made to allocate blocks
  instead - many nets at one set; he
  supernetting
- downside would be more routes i
  tables

Classless Internet Domain Rout

- pronounce “cider” -> do away wi
- allocate contiguous power of 2 bl
- represent in route table by (net, m
  where the mask indicates a range
  addresses
- think of this in simplified form as
  offset}; e.g.., net 4, +4 ([4..7] in

CIDR continued

- **routing algorithm must support prefix match for destination**
- more specific better than less spec (host route better than subnet)
- put another way, /32 better than /
- caused rewrite of most os routing algorithms and dynamic routing p
  - e.g., BGP-3 became BGP-4

thus at least two CIDR tricks

- **1. supernetting** - aggregation “up”; i.e
  from ***network*** portion
- e.g., multiples of class C nets
- **2. subnetting** (aka **Variable Length S**
  Masks); i.e., steal bits from the ***host*** p
- VLSM can be used to **subnet a subne**
  - e.g., class C assumes 24 bits of mas
  - divide that up into half (/25) and ha
    (/26)

network range notation

- express CIDR block in **slash notation**
- ip prefix/length of net mask
  - assume contiguous bits, never discontigu
- class A, thus 1.0.0.0/8
  - netmask is 255.0.0.0
- class B, thus 128.1.0.0/16
  - 255.255.0.0
- class C, 192.1.2.0/24 (255.255.2

CIDR conversion table

/15	255.254.0.0	2 class
/17	255.255.128.0	128 cl
/18	255.255.192.0	64 cla
/24	**255.255.255.0**	**1 class**
/25	255.255.255.128	1/2 cla
/26	255.255.255.192	1/4 cla
/31	255.255.255.254	1/128

VLSM can be used

- to subnet subnets, ad infinitum
- e.g., given 192.1.2.0/24
- can break up class C subnet into t
- 192.1.2.0/25 and 192.1.2.128/25
- student exercise: take 192.1.2.0/2 break it up into two /26 subnets
- this works as long as all boxes sp

examples: explain and think about

- 131.252/16 (PSU class B)
- **131.252.208.0/20** would mean what?
  - supernet or VLSM? how much of class B
- 131.252.215.3/32 (host route)
- 131.252.215.0/24 (VLSM or supernet
- 131.252.215.32/28, range is?
- 207.98.0.0/17 - how many class C add
  - range on latter is 0.0 - 207.98.127.255
- 0.0.0.0/0

assume previous slide is rout table with IP destinations

- which entry is chosen for each ds
- 1. 131.252.215.3
- 2. 131.252.215.4
- 3. 131.252.215.34
- 4. 131.252.215.0
- 5. 131.252.216.1
- 6. 131.252.1.1
- 7. 207.98.102.128, or 8. 207.98.240.1

NFS/UDP dump - decode IP

etherfind -x -v -between nfsserver nfsclient
1. UDP from server.2049 to client.1022 8300 by
1480 bytes long) (more fragments)
08 00 20 02 20 f2 00 00 3c 00 19 56 08 00 <- et
45 00 05 dc ba 65 20 00 1e 11 91 2e 8f b9 06 01
(udp, etc...)
find ip tos, len, id, flags, offset (0), ttl, proto, src

2. UDP fragment offset = 1480, length = 1480, M
(ignore ethernet, look only at IP)
45 00 05 dc ba 65 20 b9 1e 11 90 75 8f b9 06 01

Jim Binkley

NFS/UDP - IP fragmentation

3.4.5. 3 fragments, length = 1480, MF set

last fragment:
6. UDP fragment offset = 7400, length = 900
45 00 03 98 ba 65 03 9d 1e 11 af d5 8f b9 06 01

note: MF is off, id still == ba65

study questions

- decode previous NFS/UDP dump at ip layer
- construct class C subnet as asked previously
- ISP gives you two class C subnets but you ha
  wire. How do you make all hosts talk to eac
- take previous class C/VLSM example and di
  /26. what happens if you run RIPv1 on that
- refer to the previous class B subnet picture.
  mask is wrong and == 255.255.0.0 and a hos
  subnet tries to talk to another, what would ha
  dumb pc routing algorithm)?

Jim Binkley

more study questions

- what would 0.0.0.0/7 mean?
- why is 1.0.0.0/7 wrong? (think about i
  base 2 and draw a picture of the bit/s)
- say you have network 131.252.215.0/2
  break it up into 8 subnets of equal size
  - what are ranges and directed broadcast ad
- take previous 8 subnets and make it in
  subnets, 1 of which has twice the host
- make up an example where you use su