IP Basics

IP networks

IP networks are made up of

- hosts
- middleboxes (e.g., firewalls, NATs, NAPTs, Application Layer GWs)
- routers (obsolete terminology – gateways)

It will be useful to differentiate between

- core routers (connect to other routers)
- edge routers (connect to hosts)

To understand how a router is different from other network elements

we need to know the basic principles the IP protocol architecture

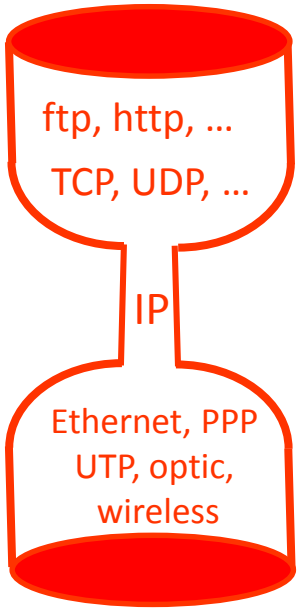
Internet - the basics (1)

The **first principle** of the Internet is the end-to-end (E2E) principle

All functionality should be implemented
only with the knowledge and help of the application at the end points

The **second principle** is the hourglass model

- *IP (l3) is the common layer*
- *below IP (L3) is not part of IP suite, above is*



Thus :

- most functionality and state is in the hosts
- middlebox functionality is severely limited
- routers are limited to forwarding packets without extensive packet manipulation (exception - TTL)

The **third principle** is that IP forwarding is

- connectionless
- on a hop-by-hop basis

Internet - the basics (2)

The **fourth principle** is that unicast IP forwarding is performed based on a **D**estination **A**ddress (DA)

- Addresses must *usually* be unique (end-to-end principle)
- Hosts usually have a single IP address, routers have many addresses
- It is the responsibility of a service provider (SP) to allocate addresses

IP addresses are not arbitrary, like Ethernet MAC addresses

The **fifth principle** is that IP addresses are aggregated into subnetworks

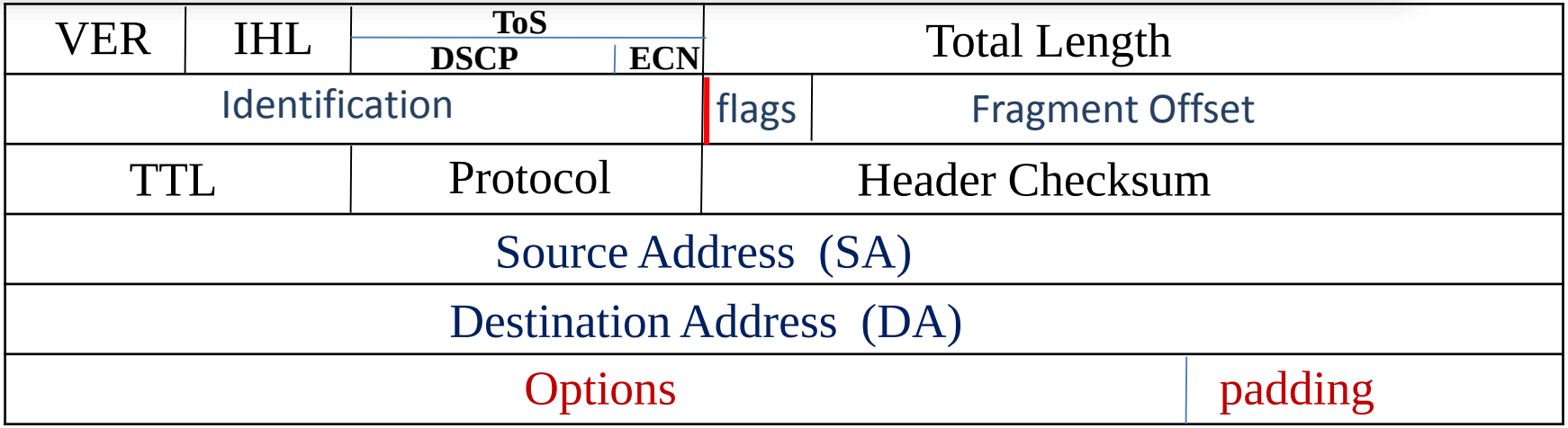
- All addresses in a subnetwork share a common prefix
- Subnetworks may be further aggregated

The **sixth principle** is that it is the responsibility of the router to forward towards the host's subnet but it is not its responsibility to deliver the packet on the subnet

- the IP suite starts above L2
- subnet's L2 (e.g., Ethernet, PPP) delivers the packet to the host

The IP Header(s)

IPv4 header



VER (4b) = 0100
IHL (4b) = Internet Header Length in 32 bit words (if no options then IHL=5)
ToS (1B) = DSCP(6b) + ECN(2b)
Total length (2B) = length of header + payload in bytes (≤64K)
Identification (2B) = ID for fragment
Flags (3b) = *reserved bit* + DF (don't fragment) + MF (more fragments)
Fragment Offset (13b) = byte number of first byte in fragment (from zero)
Header Checksum (16b) = 1s complement sum of header words (≠0)
TTL (1B) = Time To Live counter decremented by each forwarder
Protocol (1B) = protocol code (maintained by IANA, 1=ICMP 4=IPv4 6=TCP 17=UDP)
Addresses **(4B each)**

IPv4 exhaustion

1981 : RFC 791 (IPv4) allows 4.3E9 addresses
(was considered overkill at the time)

Late 1980s: IETF started predicting address exhaustion

Mid 1990s : Internet commercialization

Allocated addresses vs. time

- 1990 25%
- 1996 50%
- 2005 75%
- 31 January 2011 IANA allocated the last /8 address blocks
- 2012-2015 Regional Internet Registry exhaustion

16-bit AS number exhaustion expected in 2018 !

Although IPv6 defined in 1998 (RFC 2460) allows **3.4E38** addresses
only about 15% of global Internet traffic is now IPv6

Why is it taking so long ?

CIDR and NAT were too successful

fooled ISPs into thinking that they had lots of time !

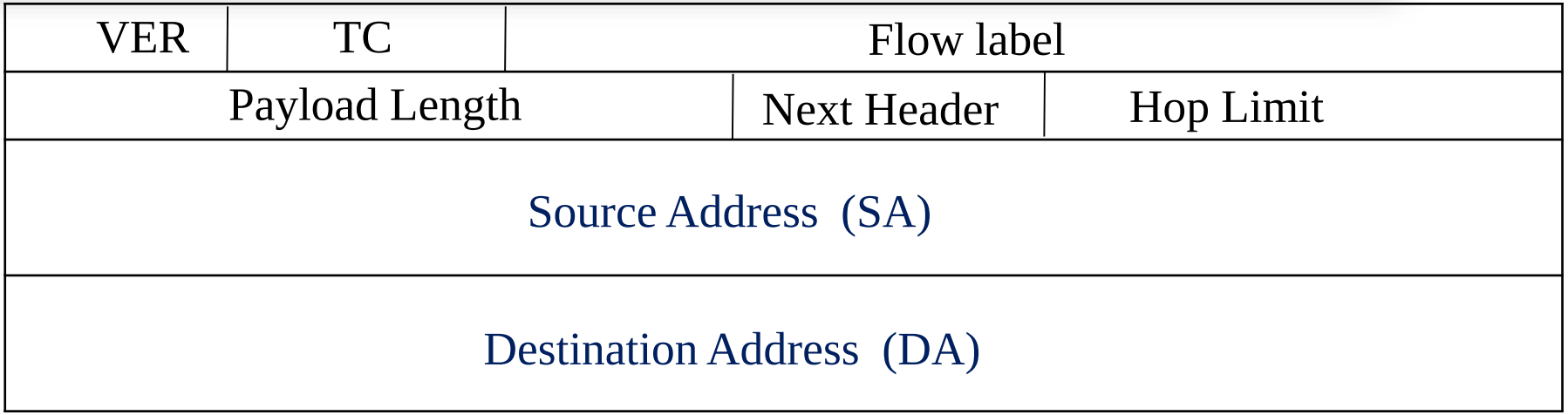
Mobile devices and always-on connections made things worse

Inefficient address allocation (IANA/RIR/registrar model)

Internet provision business model

- service is provided by a succession of providers
- end-end IPv6 requires all providers to support
- it is in no SP's best interest to migrate before all the rest
- few application servers supported IPv6 (this changed after the *launch*)

IPv6 header



VER (4b) = 0110
TC(8b) = Traffic Class, similar to IPv4 ToS (= 6b DSCP + 2b ECN)
Flow Label(20b) : if ≠ 0, hint for routers to keep on same ECMP
Payload Length (2B) = length of header + payload in bytes (≤64K)
Next Header(1B) = similar to IPv4 protocol field
Hop Limit (1B) = similar to IPv4 TTL
Addresses (16B =128b each)

Note: IPv4 allows 2^{32} ≈ 4.3E9 addresses, IPv6 allows 2^{128} ≈ 3.4E38 addresses

ToS

RFC 3168 gives the history of the ToS byte
which was redefined several times

Originally it contained :

- Precedence (3b) similar to Ethernet P-bits
- D (1b) requests low delay
- T (1b) requests high throughput
- R (1b) requests high reliability
- reserved (2b)

RFCs 2474 and 3168 specify the modern form :

- DSCP (6b) (suggested values in RFC 4594)
- ECN
  - 00: Non ECN-Capable Transport - Non-ECT
  - 10: ECN Capable Transport - ECT(0)
  - 01: ECN Capable Transport - ECT(1)
  - 11: Congestion Encountered – CE

TTL

TTL is used to
- terminate routing loops
- limit the lifetime of TCP segments
- enable traceroute

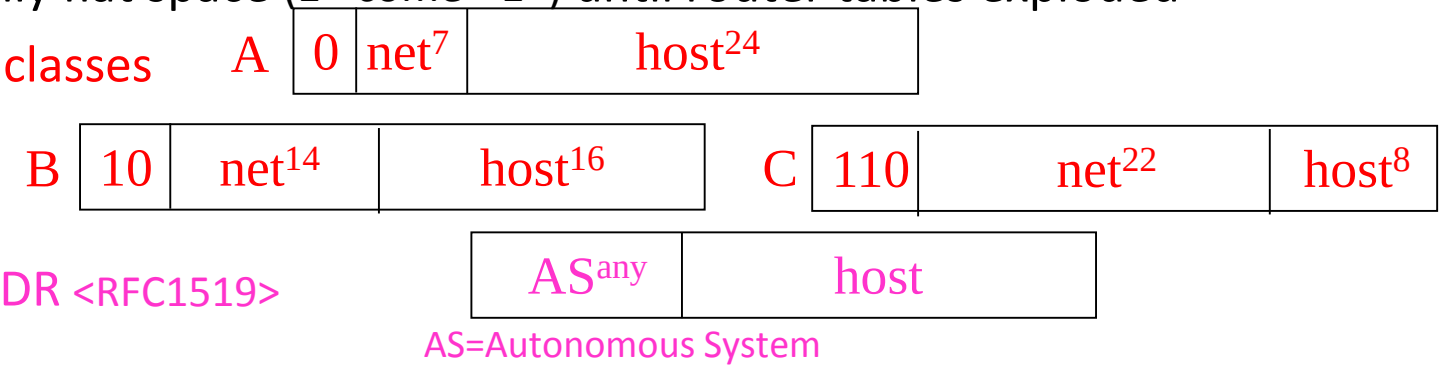
TTL was originally intended to be the time-in-flight in seconds
but since a router must decrement by *at least 1* (even if the time << 1 sec)
each router is required to reduce the TTL field by at least one
so it is effectively a hop-count
(today no-one decrements by more than 1 because of time)

TTL expiration causes *routers* to discard packets
but TTL=1 never triggers a discard for
- a destination *host*
- a router receiving a packet destined for it

IP addresses and hierarchy

IPv4

- originally flat space (1st come - 1st) until router tables exploded
- then 3 classes

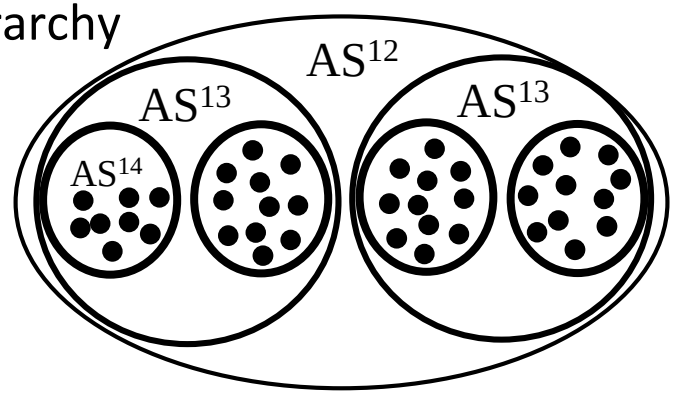


- now CIDR <RFC1519>

AS=Autonomous System

The set of all addresses with a shared prefix is called a *subnet*

IP can even support arbitrary levels of hierarchy
by advertising aggregated addresses



Addresses with lots of ones

RFC 919 defined the *IP broadcast address* FF.FF.FF.FF (all-ones)
sometimes called *all subnets* or *limited broadcast*

RFC 922 extended this to subnet broadcasting prefix+all-ones
this address is called *all hosts* or *directed broadcast*

There are several other *special* IP broadcast addresses
that must be treated in the same way as all-ones

Broadcasts must be supported by L2, and are used for purposes such as
ARP, DHCP, routing advertisements to "all routers"

Broadcast packets are never forwarded by L3 forwarding devices

All-ones must never appear as SA
and as DA must be accepted by all routers and hosts

Directed broadcasts may appear as SA
as DA must be accepted by hosts
and by default must be accepted by all routers
and by default are forwarded (only prefixes count)
but there must be an option not to forward, which must default to forward
might want to turn off for security reasons!

Addresses with lots of zeros

RFC 950 declared prefix+all-zeros to be an illegal address
RFC 1878 repealed this,
  but <network-prefix>+0 is reserved (as it once indicated directed broadcast)

However, all-zeros (0.0.0.0) is never assigned as an address
  as it stands for *this host on this network*

A router should never send a packet to 0.0.0.0
  except as part of its own initialization

A router receiving a packet with source 0.0.0.0
  must never forward it
However there are protocols that use this (e.g. DHCP, ICMP mask request)
  a router must accept such a packet if it knows the protocol

A router receiving a packet with destination 0.0.0.0
  should silently discard it
  but may treat it as a broadcast

Local loopback addresses

Network 127
(i.e. all addresses 127.X.X.X = 01111111.X.X.X)
is a network of host internal loopback addresses

Such addresses must never appear on a physical link (outside host)

A host can send a packet to itself using this kind of address
For example :
a router needing to send a packet through the forwarding engine a 2nd time or to its control plane
can address it to a loopback address

A router should not forward a packet with a loopback source address except over a loopback interface
but it may have a switch that disables this check

Multicast addresses

RFC 1112 defines IP addresses with MS byte E0- EF
(i.e. addresses 224.0.0.0-239.255.255.255)
to be multicast addresses

A multicast address stands for a group of hosts

- membership is dynamic (hosts may join/leave a group at any time)
- there is no restriction on the number of hosts in a group
- a host may belong to many groups

Multicast IP forwarding is performed by a *multicast router*
which may be a regular router too

Addresses between 224.0.0.0 and 224.0.0.255 (inclusive)
are reserved for special purposes (e.g., routing protocols)
Multicast routers should not forward packets with such a destination address

We won't discuss multicast further

Prefixes and masks

Since 1993 (RFC 1519 - CIDR) subnets can have any length prefix

There are two ways of specifying the **prefix length**

- slash notation, e.g., 192.168.16.0/20
  note – unspecified bits are set to zero
- mask notation, e.g., 192.168.16.0 with mask 255.255.240.0

Note that 192.168.16.0/20 means all addresses
from 192.168.16.0 through 192.168.31.255
Note that it contains 192.168.16.0/21, 192.168.24.0/22, etc.
since they are in the range and have longer prefixes (larger masks)

/32 are **fully qualified IP addresses**

0.0.0.0/0 matches every IP address –

- it is the default route
  route taken when there is no matching entry in FIB
- *the gateway of last resort*

Prefix tables

slash	Mask
A.B.C.D/32	255.255.255.255
A.B.C.D/31	255.255.255.254
A.B.C.D/30	255.255.255.252
A.B.C.D/29	255.255.255.248
A.B.C.D/28	255.255.255.240
A.B.C.D/27	255.255.255.224
A.B.C.D/26	255.255.255.192
A.B.C.D/25	255.255.255.128
A.B.C.0/24	255.255.255.0
A.B.C.0/23	255.255.254.0
A.B.C.0/22	255.255.252.0
A.B.C.0/21	255.255.248.0
A.B.C.0/20	255.255.240.0
A.B.C.0/19	255.255.224.0
A.B.C.0/18	255.255.192.0
A.B.0.0/17	255.255.128.0

single host route

Class C →

slash	mask
A.B.0.0/16	255.255.0.0
A.B. 0.0/15	255.254.0.0
A.B. 0.0/14	255.252.0.0
A.B. 0.0/13	255.248.0.0
A.B. 0.0/12	255.240.0.0
A.B. 0.0/11	255.224.0.0
A.B. 0.0/10	255.192.0.0
A.B.0.0/9	255.128.0.0
A.0.0.0/8	255.0.0.0
A.0.0.0/7	254. 0.0.0
A.0.0.0/6	252. 0.0.0
A.0.0.0/5	248. 0.0.0
A.0.0.0/4	240. 0.0.0
A.0.0.0/3	224. 0.0.0
A.0.0.0/2	192. 0.0.0
A.0.0.0/1	128. 0.0.0
0.0.0.0/0	**0.0.0.0** *GW of last resort*

← **Class B**

← **Class A**

Note :
for /25 D=0 or 128
for /26 D= 0, 64, 128, or 192
etc.

Some special IP addresses

Some IP addresses are reserved for special purposes and may require special treatment by router

prefix	range	purpose
0.0.0.0/8	0.0.0.0 – 0.255.255.255	defaults
10.0.0.0/8	10.0.0.0 – 10.255.255.255	private addresses
127.0.0.0/8	127.0.0.0 – 127.255.255.255	loopback addresses
169.254.0.0/16	169.254.0.0 - 169.254.255.255	link local (RFC 3927)
172.16.0.0/12	172.16.0.0 - 172.31.255.255	private addresses
192.0.2.0/24	192.0.2.0 - 192.0.2.255	Documentation
192.88.99.0/24	192.88.99.0 - 192.88.99.255	IPv6-IPv4 relay
192.168.0.0/16	192.168.0.0 - 192.168.255.255	private addresses
198.18.0.0/15	198.18.0.0 - 198.19.255.255	device benchmark
224.0.0.0/4	224.0.0.0 – 239.255.255.255	multicast
240.0.0.0/4	240.0.0.0 – 255.255.255.255	reserved

Fragmentation

All IP hosts/routers must be able to handle 576 byte packets

IP packets can be up to 64K bytes in size
but L2 may limit the size of packets that can be transported

IPv4 routers must be able to fragment an incoming packet
into a number of forwarded packets
(and should minimize the number of fragments)

All hosts must support reassembly (warning – security hole!)
and routers must be able to reassemble packets for itself (e.g., routing)

(IPv6 does not use fragmentation, instead the sender host determines MTU)

MTU may be learned from routing protocols
When originating a packet
a router should support RFC 1191 path MTU discovery

We won't discuss fragmentation further

IP Options

Some options require the router to inserts its address into the header (if there is free space)

Unrecognized IP options must be passed unchanged

Routers MUST support

- Source Route options (but there may be an option to discard such packets)
- Record Route option
- Timestamp option

Packets with IP options are often forwarded via “slow path”

Some large routers are rumored to discard any packet with options

We won’t delve into option processing

Routing

Routers

A router is a combination of hardware, software, and memory
that is responsible for *forwarding* packets towards their destinations

Routers generally work at ISO layer 3 (network layer)
but can also function at layer "2.5" (for MPLS)
and may inspect higher layers, but only for *optimization*
(QoS management, load balancing, etc.)

router 

Note that Ethernet switches technically *filter* rather than *forward*

switch 

In order to correctly fulfill their function (i.e., to know where to forward)
routers usually run *routing protocols*
to exchange information between themselves

Ethernet switches do not need such protocols as they *learn* how to filter

So the router performs 2 distinct algorithms :

- forwarding algorithm (forwarding component)
- routing algorithm (control component)

What does a router do ?

Control plane (routing algorithm)

- run routing protocols
- identify interface and next hop L2 addresses
- populate RIBs (if Link State, perform SPFs)
- scan all RIBs, and produce FIB (entries map FEC to NH)

Data plane (forwarding algorithm)

- deframing (CRC/checksum/defragmentation/reassembly/demapping…)
- parsing (pulling values from appropriate fields – simple IPv4 DA, complex finding URL or MIB variable)
- FEC classification (add metadata, based on DA, DA+ToS, MPLS, …)
- lookup / search
- packet modification and replication
- framing
- traffic management and queuing
- compression, encryption, etc.

Router interfaces

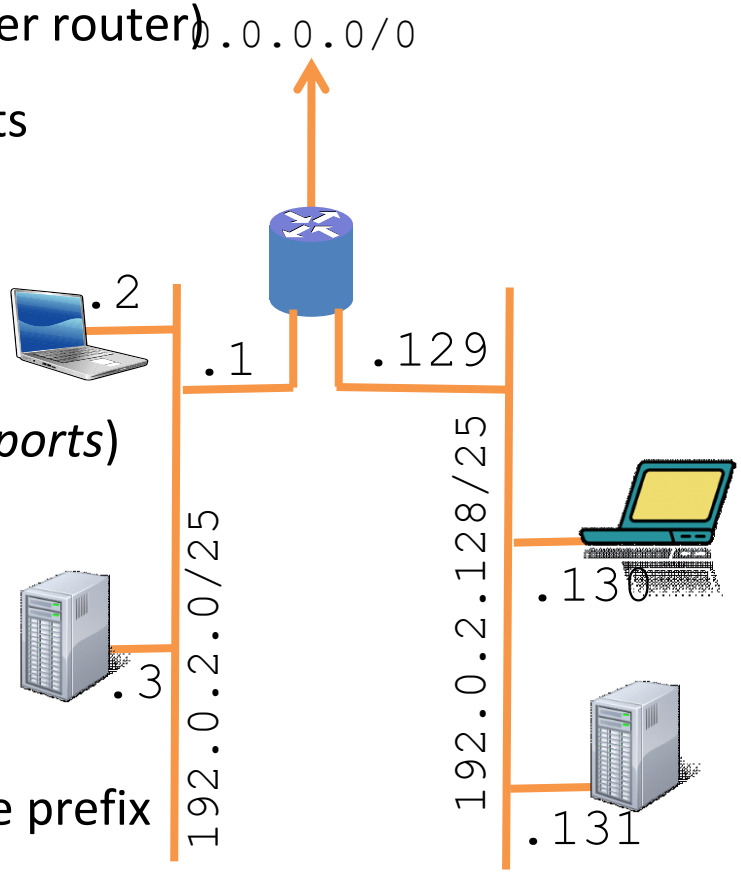
Routers connect to hosts and to other routers via *interfaces*
(from 1 to many thousands of interfaces per router)

Routers are responsible for forwarding packets
- arriving at an *ingress* interface
- to an *egress* interface

Interfaces have layer 3 and above properties
and also contain layer a and 2 properties (*ports*)

Most interfaces have unique IP addresses
(but there are also un-numbered interfaces)

Interfaces are grouped into subnetworks
All interfaces on a subnetwork share the same prefix



Router forwarding

Based on the 6 principles we can understand how routers forward

1. The router looks at the packet header
2. It deduces to which subnet the packet belongs
3. If the router can directly interface that subnet
   it must use the appropriate L2 to send the packet to the host
4. Otherwise it must retrieve the next hop (router)
   that sends the packet towards the subnet
5. The next router does the same
6. If routing has converged there will be no *loops* or *black holes*
   but there *may* be during transients

The information needed by the router to properly forward packets
is stored in the **F**orwarding **I**nformation **B**ase (FIB)

The FIB associates address prefixes with **N**ext **H**ops (NHs)
(and, to save an additional lookup, usually with L2 addresses as well)

Do not confuse the FIB with a **R**outing **I**nformation **B**ase (RIB)

More on FIBs

Simple (and primitive) routers have a *routing table*
modern large routers have several different databases

- The FIB is designed to be fast to search
- RIBs are designed to be fast to update

There may be many RIBs, one for each routing protocol running
and static routes may be entered into any of them

There are sometime other databases as well
for example – link state routing protocols require a LSDB
from which the RIB is built

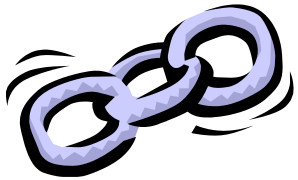
The FIB is built from RIBs

FIB

entry	prefix	interface	NH	L2	L2 parameters
0	192.168.16.0/24	1	10.10.1.1	PPP	…
1	192.168.196.0/20	2	10.16.54.2	Eth	…
2	192.168.0.0/17	2	10.16.1.16	Eth	…
3	0.0.0.0/0	3	10.1.1.0	Eth	…

IP Routing types

- Distance Vector (Bellman-Ford), e.g. `RIP, RIPv2, IGRP, EIGRP`
  - send <addr,cost> to neighbors
  - routers maintain cost to all destinations
  - need to solve "count to ∞ problem"
- Path Vector, e.g. `BGP`
  - send <addr,cost,path> to neighbors
  - similar to distance vector, but w/o "count to ∞ problem"
  - like distance vector has slow convergence*
  - doesn't require consistent topology
  - can support hierarchical topology => exterior protocol (EGP)



- Link State, e.g. `OSPF, IS-IS`
  - send <neighbor-addr,cost> to all routers
  - determine entire flat network topology (SPF - Dijkstra's algorithm)
  - fast convergence*, guaranteed loopless => interior routing protocol (IGP)

*convergence time is the time taken until all routers work consistently
before convergence is complete packets may be misforwarded, and there may be loops

IDR

What is the relationship between all these routing types ?

The Internet is composed of **A**utonomous **S**ystems run by network operators

Each AS is truly autonomous

- AS is a single entity to the outside world
- routers in the same AS obey a common policy, and *trust each other*
- one AS can *request* another to forward a packet, but can not *force* it to

Inside an AS topology information is shared

- Link State routing (OSPF, IS-IS)

Between AS's topology information is on a need-to-know basis

- Path Vector routing (BGP)

Each AS has at least one **AS B**order **R**outer

- one leg inside the AS (OSPF/IS-IS)
- one leg between AS's (BGP)
- transit AS has at least 2 ASBRs
- ASBR application uses policy to decide what to advertise for IDR
- peering relationships influence policy

More of the story

Actually, it can get a lot more complicated

In general, a router will be running multiple routing protocols

For example :

- one or more IGPs (RIP,OSPF, IS-IS) between routers in the same AS
- **i**nternal BGP (iBGP) between routers in the same AS
  (usually a full mesh, but when too complex we can use route reflectors)
- **e**xternal BGP (eBGP) between ASBRs in different ASs

How does a router know if a BGP session is iBGP or eBGP ?

- by the AS number !

IGP is used to find a path to another router (including ASBR) in the same AS

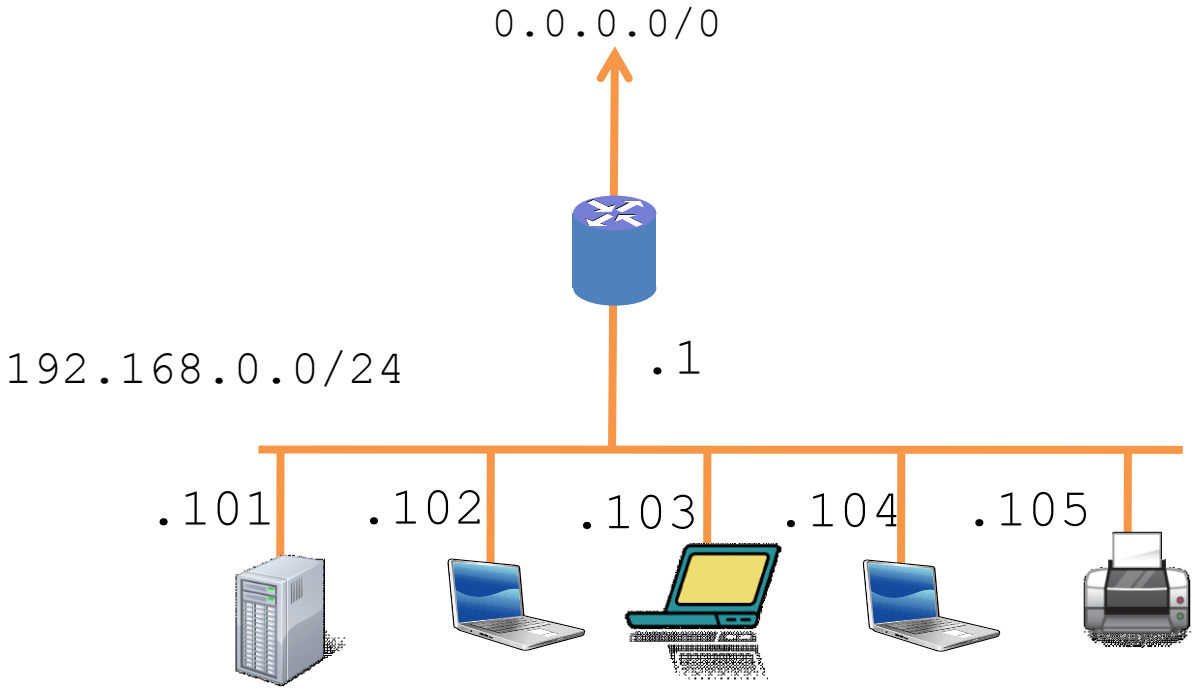
eBGP is used by ASBRs to learn / distribute routes to other ASs

iBGP is used for ASBR to inform core routers of external routes

Simplest example

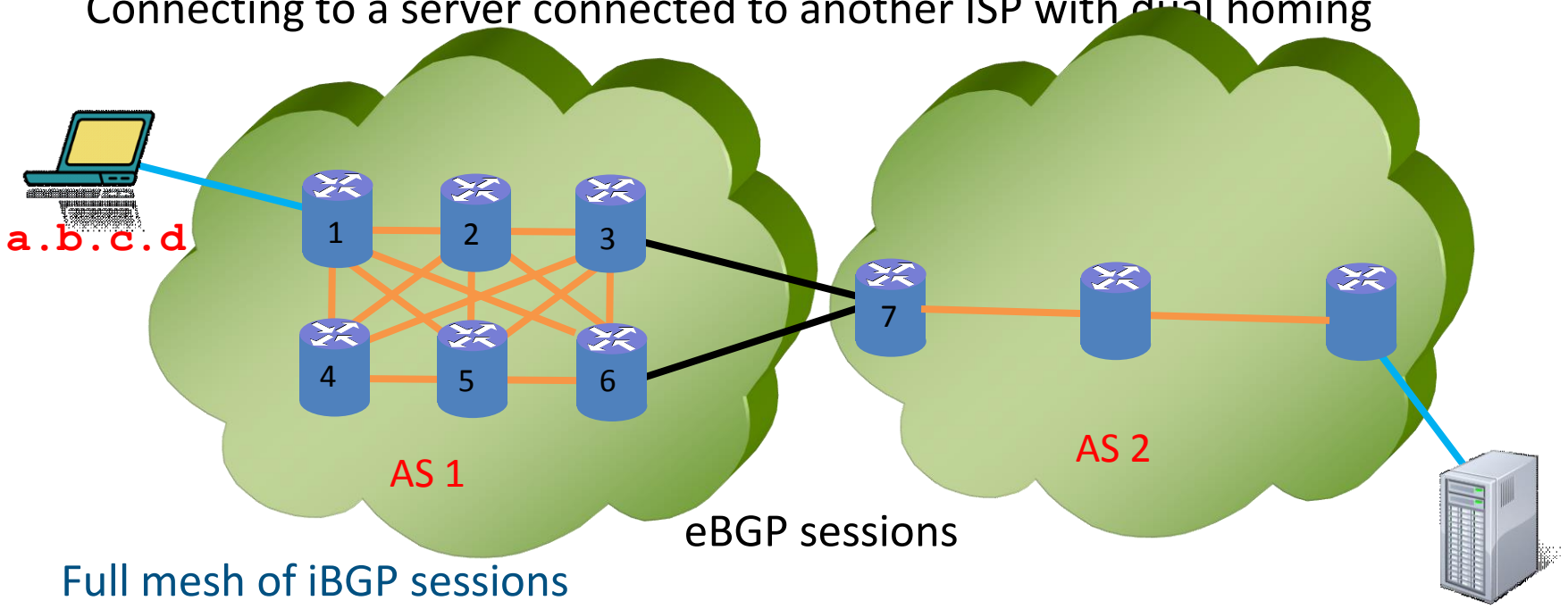
Stub ASs (my home router)

- single homed to outside world
- single internal subnet, so don't need IGP
- single homed, so don't need to run BGP to ISP
- don't need to have an AS number



More complex example

Connecting to a server connected to another ISP with dual homing

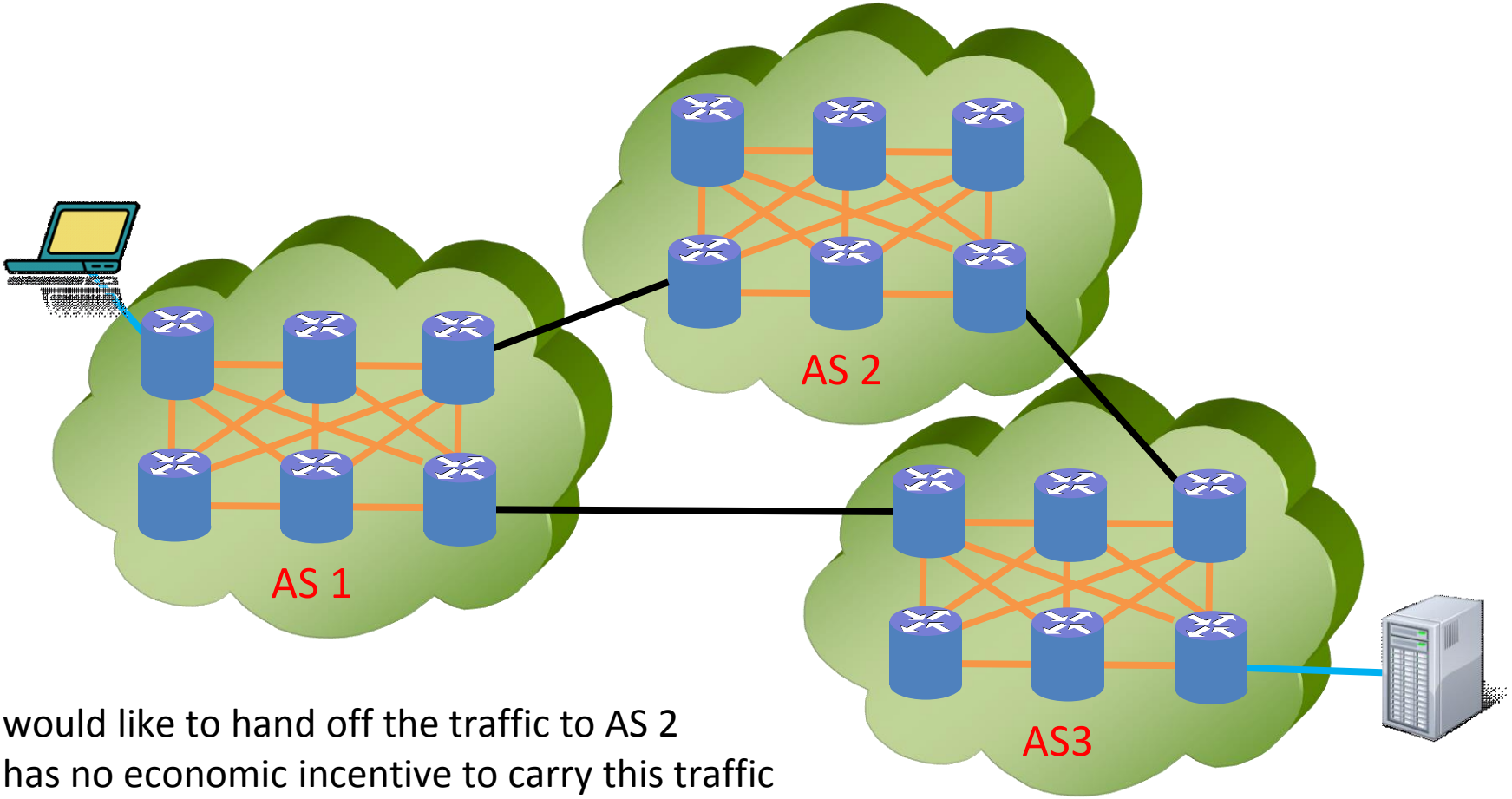


Full mesh of iBGP sessions

- Routers 3 and 6 learn from eBGP how to reach A.B.C.D
- Policy determines that 3 will be used (see later)
- Router 1 learns from iBGP session that A.B.C.D is reachable via router 3
- Router 1 learns from IGP that router 3 is reachable via router 2
- Router 2 knows how to directly reach router 3 because of IGP adjacency
- Packet from a.b.c.d is forwarded via 1-2-3 to AS 2 and to A.B.C.D

Even more complex example

Three ASs, with one possibly acting as a transit domain



AS 1 would like to hand off the traffic to AS 2
AS 2 has no economic incentive to carry this traffic
But AS 2 gets the route from AS3
What can AS 2 do to stop this ? (remember autonomy!)
What can AS 1 then do ?

Rules for customer ASs

Stub AS

Single-homed AS does not need to learn routes from provider
It only has to send all traffic via its unique exit point (0.0.0.0/0)
Provider gets routes from static or IGP or private-AS eBGP

Multihomed Nontransit AS

AS advertises only its own routes to both SPs
AS filters out traffic for foreign routes that reach it via static/default routing
eBGP is not needed, but recommended for route propagation and filtering

Multihomed Transit AS

Uses eBGP to SPs and iBGP for transit traffic

BGP rules

There are :

- internal routes
- external routes
- customer routes

When eBGP learns a route it is repeated via iBGP to all others in AS
thus all routers in AS learn it

When iBGP learns a route it is repeated only to externals via eBGP
since internals also get it directly

When there is another ASBR that can reach the same other AS
a second route is repeated by iBGP to the ASBR

The ASBR will then make the decision as to which to use !

IGP rules

IGPs are used between routers in the same AS
- so IGPs do not have sophisticated policy control
- routers usually blindly accept all information received

For proper operation (no routing loops)
- all routers in AS must have the same IGP RIB
- for link state protocols (OSPF, IS-IS)
  there is a **L**ink **S**tate **D**ata **B**ase (LSDB)
  from which IGP RIBs can be constructed (will be explained shortly)

Because all routers have the same LSDB

Although the forwarding is hop-by-hop
  the result is the same as if there were coordination

IGPs
- do not scale to infinity
- require complete knowledge
- are not suitable for interaction with non-trusted routers
  since a single misconfiguration can be fatal

LSDBs

We said before that LS routing protocols have another database

LSDB contains representation of every router and link in the AS
implicitly holding the complete *topology* of the network

In addition, the LSDB associate costs (metric) with every link

- RIP - the metric is always hop count, no non-trivial metric
- OSPF - the metric is more general, for example link length

These costs form a matrix

From \ to	Router A	Router B	Router C
Router A		M(A,B)	M(A,C)
Router B	M(B,A)		M(B,C)
Router C	M(C,A)	M(C,B)	

The topology is symmetric, but the costs need not be

LSDBs and IGP RIBs

Each router can independently calculate the least-cost path
to every other router in the AS

A **S**hortest **P**ath **F**irst (SPF) algorithm (e.g., *Dijkstra's algorithm)*
is used to compute a tree of the shortest paths to all destinations

Each route in the SPF tree is an entire path
but for each router we can extract the next hop
and build the RIB for that router (each router has its own RIB)

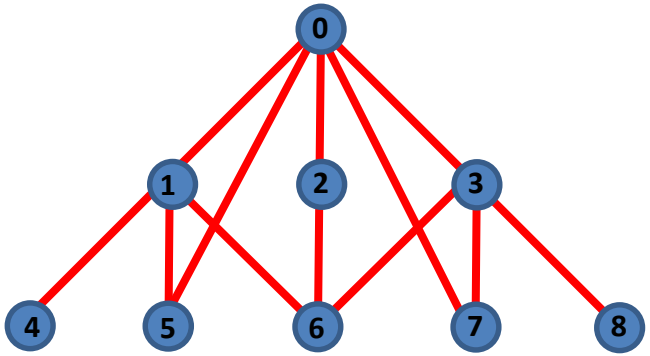
From the RIBs we build the FIB needed for efficient forwarding of packets

Graph search algorithms

There are many algorithms for search on graphs :

- Breadth first
  - Bellman-Ford
  - Iterative deepening
- Depth first (backtracking)
  - Depth limited
- Best first
  - Greedy algorithms
    - Dijkstra's algorithm
  - Beam search
  - A*
  - B*

etc. etc.



Dijkstra's algorithm

Graph search algorithm first described by Edsger Dijkstra in 1959
It assumes additive, non-negative, costs for each link in graph
It is a best-first *greedy* algorithm

Think of a city street map
We want to from initial intersection to destination one with the least walking

Start at the initial intersection – its distance is zero

Measure and label the distances to all adjacent intersections (breadth first)
Choose the closest one (this is the *greedy* step)

Consider all the neighbors of the chosen intersection
If the distance (sum of the distance to the chosen intersection and the distance from chosen intersection to neighbor) is the *shortest* known way to get to that neighbor
then remember that distance *(not a tree!)*

Once you have considered all neighbors of the intersection
mark the chosen intersection as visited (*its* distance is now known)

Choose the unvisited intersection with shortest distance

Continue until all intersections have been visited

Dijkstra's algorithm - formal

Let's call the node we are starting with an **initial node**.
The COST of node X will be the distance from the **initial node,**
i.e., the sum of distances of all links along the path from the initial node to X

Initialization
Set initial node's COST to zero, all others to infinity
Set initial node as current, all other as unvisited

Main step
For all unvisited neighbours of current node :
Calculate their distances from the initial node as
COST(neighbor) = COST(current) + DIST(current to neighbor)
If this is less than what is presently marked, overwrite the marking

When all neighbors have been considered, mark current node as visited
(once visited, this node's COST is final)

Recurse
Select the unvisited node with the smallest COST as current node
Go to Main step

Implementation issues

If our graph has N nodes and L links

In a straightforward implementation of Dijkstra's algorithm

- finding the unvisited node with lowest cost takes O(N)
- this is done N times
- so the total computation for this is $O(N^2)$
- the computation of the distances to every node takes O(L) since each link is followed once (to the node it lands on)

So the total complexity is $O(N^2 + L) \approx O(N^2)$

By using more sophisticated data structures (Fibonacci heap) this can be reduced to $O(N \log N + L) \approx O(N \log N)$

RIBs to FIB

So we are *finally* ready to see how the FIB is populated

First rejection rules are applied, for example :

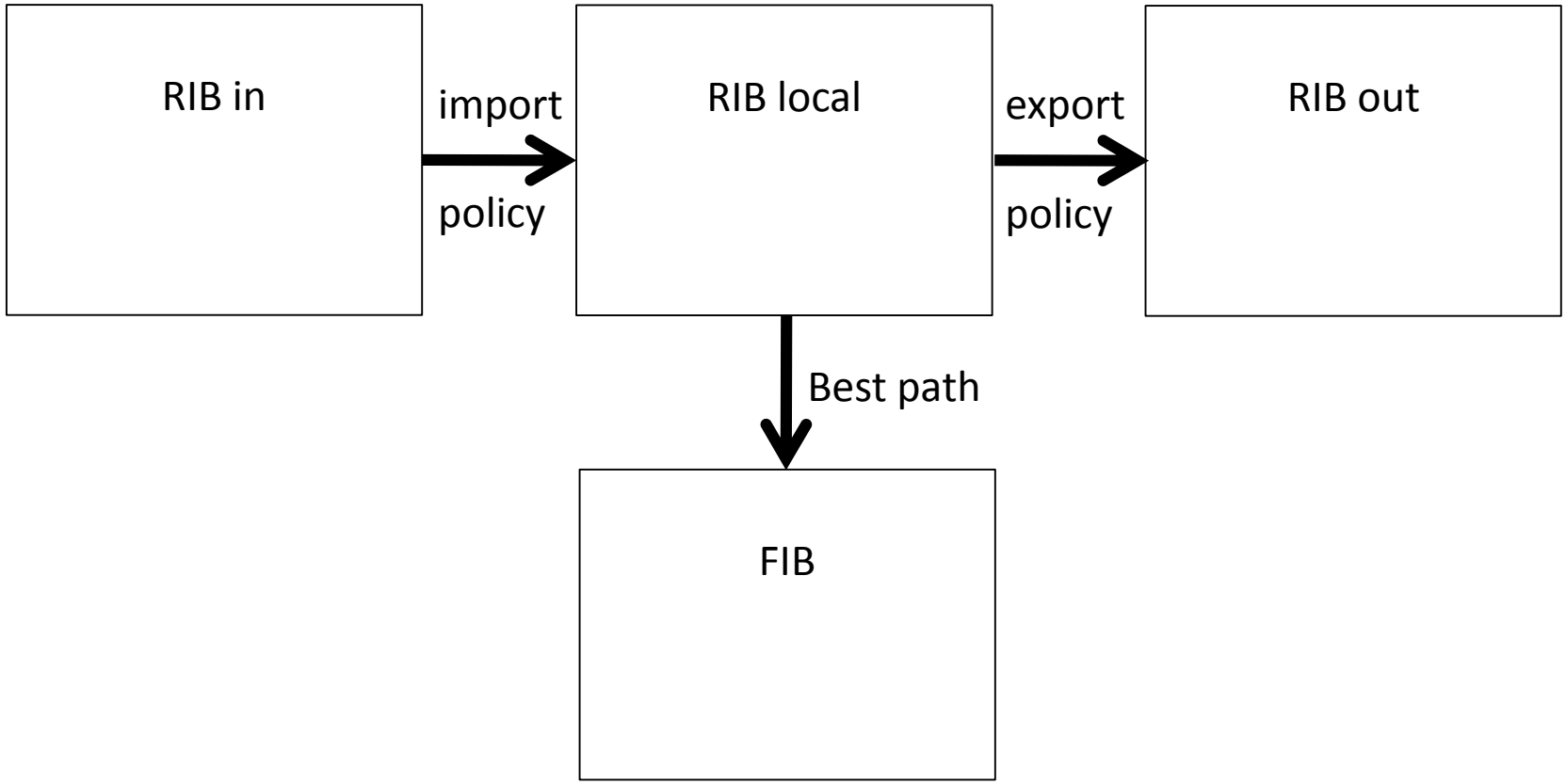
- do not accept routes from ASs without agreements
- do not accept routes that loop
  (e.g., BGP advertisements with AS number in the AS-PATH)

Then install FIB entries according to policy, for example :

1. first install Static routes
2. then routes from IGP RIB
3. choose eBGP before iBGP
   (hot potato rule- get it out of my network - let someone else handle)
4. if there are different routes from BGP
   choose the route with highest local preference
5. if routes have equal local preference
   choose the route with the shortest AS-PATH
6. if routes have equal AS-PATHs
   choose the route with the lowest origin number
7. if still equal choose highest BGP peer address

Advertisement

Not everything received is accepted for inclusion in the FIB
Not everything accepted for inclusion in the FIB is further advertised
Never advertise information not accepted to FIB !



Forwarding Equivalence Classes

Forwarding Equivalence Classes

Simple routers typically search for IP prefix
and immediately perform forwarding
e.g., the UNIX router LSDB Dijkstra returns forwarding information

Today's routers decouple (as much as possible)

- routing protocol(s) (building LSDB, RIBs)
- building FIB
- packet parsing/classification/search
- forwarding decision

All packets that are to be forwarded in the same way
are grouped together to form a FEC

Thus the search algorithm returns a FEC label
and then a second lookup is performed
to find the forwarding information

Equivalence Classes

In *set theory* we define an **E**quivalence **C**lass as:
set of elements that can be considered equivalent for some purpose

Theorem
Any equality relation (e.g., common features) divides elements into non-overlapping equivalence classes

reflexive: **a ~ a**
symmetric: **a ~ b ⇨ b ~ a**
transitive : **a ~ b and b ~ c ⇨ a ~ c**

Example:
equality modulo 3 for positive integers
A = B (mod 3) if and only if A = 3a + c and B = 3b + c
or (A-B) divides by 3
there are three equivalence classes :
{ 0, 3, 6, 9, 12, … }
{1, 4, 7, 10, 13, …}
{2, 5, 8, 11, 14, … }
note that every positive integer is in exactly one EC

Forwarding Equivalence Classes

A FEC is the set of all packets that are to be treated in the same way

By the theorem every packet belongs to one unique FEC

So the router's forwarding job is now :

1) parse packet
2) search and classify packet as belong to a particular FEC
3) forward based on FEC's forwarding information

Packets in the same FEC should follow the same path
but in IP this is not directly *enforced*
since each successive router reclassifies the packet's FEC

If the router could insert information into the packet informing the next router of its FEC

- this would save a lot of processing at the following routers
- the subsequent forwarding would be CO instead of CL

Unfortunately, this is impossible (without label switching)!

FECs

What constitutes a FEC ?

For plain IP routing

all packets w/ same destination IP prefix

that prefix being the longest in the routing table

We would like more control

- coarsest granularity

  all packets with a destination address

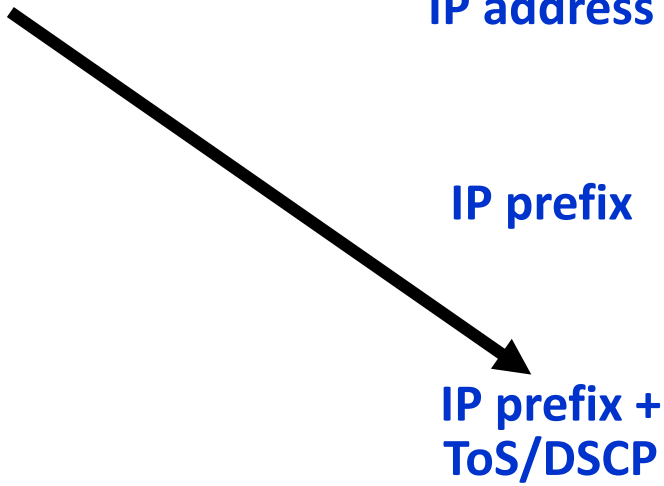
  served by a given router

- finest granularity

  all packets from given source socket

  to given destination socket

  with specified handling requirements



Requirements for Routers

Routers

A *router* is an IP network element with interfaces on at least 2 subnets

Routers were originally *gateways*

(the term is still seen in default GW, Border Gateway Protocol, …)

although now that word is now usually reserved

for network elements that interfaces networks of different technologies

Initially there was no separation between

- forwarding (data plane), and
- routing (control plane)

this was changed by introduction of the concept of a FEC

Routers perform many functions:

- L2 functions
- IP forwarding
- IP routing (control plane)
- system support (management plane, error logging, etc.)

We normally differentiate between

- the fast path (simple forwarding)
- the slow path (control protocol packets and special cases)

Routers – IP forwarding

The forwarding plane receives and forwards IP packets

- parsing (pulling values from appropriate fields – simple IPv4 DA, complex finding URL or MIB variable)
- dropping invalid packets
- lookup/search
- classifying FECs (add metadata, based on DA, DA+ToS, MPLS, …)
- modification and replication of packets
- recognizing error conditions and generating ICMP messages
- fragments packets when necessary
- choosing a next-hop destination for each packet
  based on information in its **F**orwarding **I**nformation **B**ase (FIB)
- forwarding packet
- traffic management and queuing
- compression, encryption, etc.

*We won't discuss **all** of these …*

Routers – IP routing, etc.

Routers usually support an interior gateway protocol (IGP)
and edge routers support an exterior gateway protocol (EGP)

The control plane of a router consists of :

- run routing protocols
- identify interface and next hop L2 addresses
- populate RIBs (if Link State, perform SPFs)
- scan all RIBs, and produce FIB (entries map FEC to NH)

Routers provide network management and system support facilities, including SW uploading, debugging, status reporting, exception reporting

Routers may support BFD to monitor continuity with other routers

Routers are usually required to have high performance and availability

Error messages

IP networks are best effort,
so there are no guarantees that packets sent will actually be delivered

ICMP (in addition to the well-known ping and traceroute)
provides a basic error reporting mechanism :

- Destination Unreachable Messages
- Source Quench Messages
- Time Exceeded Messages
- Redirect Messages
- Parameter Problem Messages

There are no such reports in cases where we will say “silently discard”
and when the problematic packet is any of the following :

- an ICMP error message
- a packet that fails the header validation tests (unless specified there)
- a packet destined to an IP broadcast or IP multicast address
- a packet sent as a L2 broadcast or multicast
- a packet with a Martian SA
- a packet is a non-initial fragment

IP forwarding walkthrough

Where is this defined ?

The following process is based on IETF standards :

IPv4 is defined in STD-0005, which includes

- RFC 791 Internet Protocol (IPv4)
- RFC 792 ICMP
- RFC 919 Broadcasting Internet Datagrams obsoleted
- RFC 922 Broadcasting Internet datagrams in the presence of subnets
- RFC 950 Internet Standard Subnetting Procedure
- RFC 1112 Host extensions for IP multicasting

Behavior of IP hosts in RFC 1122

Behavior of routers in RFC 1812

IP forwarding MIB is in RFC 4292

Other details are purposely distributed among many RFCs and elsewhere

We will skip over numerous special cases and rare protocols

We will cover IPv4 only (IPv6 requires another talk)

Layer-2 delivery

IP packets are always delivered over a lower layer (L2), such as

- Ethernet
- PPP
- PoS
- GFP over serial
- GRE
- MPLS

L2 functionality includes :

- encapsulating and decapsulating IP packets into L2
- CRC checking/generation
- sending and receiving IP packets (up to MTU)
- translating IP destination address into L2 address (including ARPs)

The L2 is not specified by IP standards,
but must provide the following services :

- delivery only of packets that passed basic error detection
  i.e., discard of faulty packets
- delivery only of packets identified as IPv4
  e.g., by EtherType or UPI
- delineation of packet
  including elimination of padding
- supplying length of IP packet (we will call this the L2-length)

Initial sanity checks

RFC 1812 Section 5.2.2 mandates 5 initial *header validation* checks
checks MUST NOT be disabled, and SHOULD be performed in order

1. L2-length must be ≥ 20 B (minimum IP packet size)
2. Header Checksum must be correct
3. VER = 4
4. IHL ≥ 20 B
5. Total Length ≥ IHL

SHOULD perform the following too

6. L2-length ≥ Total Length

Note – no field is used before it is *verified*

If any check fails – MUST discard the packet

Note: If pass 2 and 3 then MAY respond with ICMP Parameter Problem message

The router MAY try to determine why the check failed

- IP header was truncated by lower layer
- IP header was corrupted
- not IPv4 (e.g., IPv6)
- sender purposely generated illegal IP header

What's next ?

Now that we are reasonably sure that the packet header is OK
we can look at it

Note that we do NOT check the TTL field yet
because packets for the router itself
are not discarded because of TTL expiry

Note that reassembly is not performed
(and is only performed by a router for packets destined for itself)

Most of the IP options are processed now (but we won't go into the details)
except those requiring the router's IP address
that can only be processed after the forwarding decision

The next step is to observe the address fields

Martian Addresses

Illegal IP addresses are called Martian addresses
because they appear *as if from outside this world*

This includes addresses previously described, for example :

- 0:0:0:0 as SA or DA
- 127:X:X:X as SA or DA
- FF.FF.FF.FF as SA
- special broadcast addresses
- multicast addresses as SA

A router should silently discard packets with Martian addresses

There may be a switch to disable this check
but it must default to “perform checks”!

When a packet is discarded because of these rules,
the details should be logged

Source Address Validation

In addition to testing for Martian source addresses,
routers should implement *source address validation*

However, this check is not enabled by default

The check entails looking up the packet's source address in the FIB
and verifying that it is consistent with the logical interface

If enabled, a router must silently discard any packet
that arrives on a logical interface
to which its address would not have been forwarded

This may be an important security provision

ACLs

Access Control Lists offer an additional (basic) security mechanism
as well as a method of controlling/limiting traffic

Routers should implement configurable ACLs

When enabled, the router observes source and destination addresses
and optionally other fields (e.g., protocol field, L4 ports)
before forwarding packets

Forwarding may be either according to include lists or exclude lists

- Include list – description of packets to be forwarded
- Exclude list - description of packets to be blocked

When a packet is blocked based on ACLs
the details should be logged
an ICMP unreachable message should be sent (configuration option)

Certain vendors have considerably expanded the ACL functionality

Where does it go ?

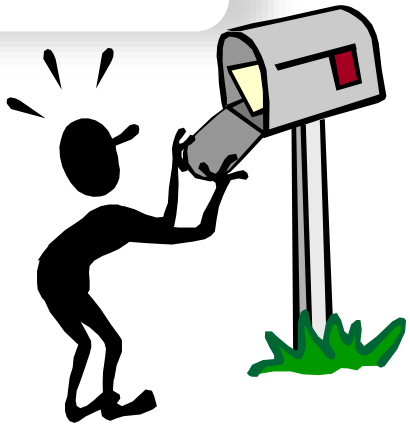
The next step is to look at the DA

There are three possibilities :

1. the packet is destined for the router (local delivery)
   and should be queued for local delivery (reassembled if needed)
   and processed according to regular IP host rules (RFC 1122)
2. the packet is not destined for the router
   and should be queued for forwarding
3. the packet should be queued for forwarding
   and a copy must be queued for local delivery

Cases where the packet is destined for the router
must be handled first

Is it for me ?



The packet is destined *only* for the router (case 1)
if its DA is any of the following :

- one of the router's addresses (exact match)
- the limited broadcast address (FF.FF.FF.FF)
- a multicast address that is never forwarded (e.g., 224.0.0.1 or 224.0.0.2)

  AND

  at least 1 logical interface associated with the physical interface

  on which the packet arrived is a member of the destination multicast group

When a router receives such a packet
it must perform all the functionality of a regular host
(including L4 functions and higher)

Is it also for me ?

The packet is destined for the router and to be forwarded (case 3)

if its DA is any of the following :

- a directed broadcast address that addresses at least one of the router's logical interfaces but does not address any of the logical interfaces associated with the physical interface on which the packet arrived
- a multicast address which may be forwarded and at least one of the logical interfaces associated with the physical interface on which the packet arrived is a member of the destination multicast group

Note :

A packet is delivered locally if the packet's DA is a directed broadcast address that addresses at least one of the logical interfaces associated with the physical interface on which it arrived
It is *also* forwarded unless the link on which the packet arrived uses an encapsulation that does not encapsulate broadcasts differently than unicasts (e.g., by using different Link Layer destination addresses).

Caveat

Routers generally do not forward packets received as L2 broadcasts

A packet is delivered locally if the packet's DA
is a directed broadcast address that addresses at least one of the logical interfaces associated with the physical interface on it arrived
It is *also* forwarded unless the link on which the packet arrived uses an encapsulation that does not encapsulate broadcasts differently than unicasts (e.g., by using different Link Layer destination addresses)

The idea is to deal with a directed broadcast to another network prefix on the same physical link. If the sender sends the broadcast to the router as a L2 unicast this is OK, since the router sees a unicast destined for a different network prefix than the sender sent it on. So the router can safely send it as a Link Layer broadcast out the same physical link. But if the router can't tell whether the packet was received as a L2 unicast, we must ensure that it plays it safe.

TTL

OK, so the packet needs to be forwarded

the next step is to decrement the TTL

if the router is so slow that the time is longer than 1 second
  the router may decrement by more than 1

if TTL $\leq$ 0 then
  the packet is discarded

  if the destination is not a multicast address

    the router MUST return an ICMP *Time Exceeded* message

a router MUST NOT discard an IP unicast or broadcast packet with TTL>0
  even if it is sure that another router along the path to the destination
    will decrement the TTL to zero

however, a router *may* do so for IP multicasts (for efficiency reasons)

Forwarding

OK, so the packet still needs to be forwarded

Now we have to determine --- to whom ?

This is decided based on the destination address

This decision isn't so simple :

- The destination address is not necessarily already in the DA field
- Once the true destination address is known
  the router must still determine
    if the destination host is directly connected to it
    and if so on which interface ?
  OR
    if it needs to pass it through another router,
    and if so, what is the next router's address (Next-Hop address)

Source routing

Finding the true destination address is made more complex
because of *source routing*

Source routing allows (partial or complete) path specification
rather than relying on the route determined by the routing protocols

Source routing adds determinism
and may enable achieving performance goals

Two different IP header options are available :

- SSRR **S**trict **S**ource and **R**ecord **R**oute
- LSRR **L**oose **S**ource and **R**ecord **R**oute
  Note: LSRR packets are often blocked since they enable spoofing attacks

Source routing places a sequence of IP addresses in the options
(intermediate routers, and the ultimate address being the host)
and the next router in the sequence in the DA

The router must use the IP DA
not the address of the ultimate destination (last address in the option)
when determining how to handle a packet

Forwarding with source routing

If the packet has more than one source route option
then the packet is discarded and an ICMP message is returned

If the packet has a source route option, and

the destination address is one of the router's addresses, and

the pointer in the Source Route Option does not point past the option end

then the next IP Destination Address is the address pointed at

If the pointer points past the end of the option the router
an error message is generated

This IP destination address is now used
as the destination address to be searched

Directly accessible ?

OK, now we have the destination address,
what do we do with it?
The next step is to determine if the destination host is directly accessible

The first step in the general algorithm is applicable
only if the router has interfaces without IP addresses

We will omit this case

The router now looks at each of its interfaces
each of which has an IP address starting with a prefix

For each such prefix

- compare to the corresponding set of bits in the packet's DA
- if they match the packet can be transmitted through the interface

Note: there can never be >1 match in a properly configured router

If no match is found, then we need to find the Next-Hop router

We omit the case where NHRP is used

Finding the next hop

OK, we now know that the host is not directly accessible
so we need to send through another router

We need to find for this next-hop router

- its L2 address
- on which interface it can be reached
- how to format the IP packet

All of this information is in the appropriate FIB entries

As was explained in the routing course, the FIB is built using

- RIBs derived from the routing protocol(s)
- static routes and default routes
- metrics
- policy

Here the question is how to find the right FIB entry

Searching the FIB

next-hop selection is performed by searching the FIB entries
and selecting the best route (if there is one)

All FIB search algorithms start out with the entire FIB and prune

Hopefully, at the end of the pruning exactly one route remains
- if none remains (including no default route)
  - the destination is unreachable.
- if many remain, the router may choose
  - based on administrative preference
  - in order to optimize metrics
  - arbitrarily
  - to perform load balancing (usually least recently used)

There are several FIB search algorithms, depending on
- traffic type (unicast, unicast with ToS, IP multicast, …)
- network configuration
- router architecture

How ToS changes things

When ToS is supported :

- router must maintain a ToS (DSCP) value for each route in the FIB
- if routing protocol does not support ToS
  packet's ToS is set to zero (default)

We will describe the default ToS forwarding algorithm

There are other proposals (similar to Ethernet QoS), such as

- low delay packets placed at head output queue
- obey drop precedence

but these are not presently required or widely implemented

When unreachable, the ICMP error codes are different with ToS

ToS forwarding algorithm

```
router retrieves all routes to destination
if none
    packet dropped (unreachable)
else
    { one or more routes }
    router performs exact match on ToS (DSCP)
    if one match
        router forwards to it
    else
        if > 1
           router forwards to destination with lowest metric
        else
           { no match }
            router looks for route with ToS=0 (DSCP=0)
            if found
             forward to it
         else { still no match }
                 packet dropped (unreachable)
```

Finishing up

OK, now we finally know which FIB entry to use

1. retrieve from the FIB the (logical) interface
   through which we need to send the packet
2. if the packet has IP options that require the router's IP address
   (e.g., Record Route, Timestamp)
   we can now process them
   using the router's address corresponding to the interface
3. fix the IP checksum
   (in general we do not have to recalculate – we can fix)
   (NB TCP checksum is never changed
   when using source routing the TCP checksum pseudoheader has the *ultimate DA*)
4. retrieve the L2 parameters for the interface
   (e.g., Ethernet MACs and VLANs, GRE parameters, …)
   and build the L2 frame
5. finally, queue the packet for transmission