

## Routing

## Basic Idea


- Routing table at each router/gateway
- When IP packet comes, destination address checked with routing table to find next hop address
- Questions:
  - Route by host or by network?
  - Routing table: how to build and maintain it?
  - Monolithic routing table structure/maintenance method across the Internet?

## Routing by the Network


- Routing by host address – explodes routing table at higher level routers
- Subnetting – allowed better management of IP space, but by itself, still needs one entry per subnet at higher level routers
- **CIDR** (Classless Inter Domain Routing)
  - Combine contiguous networks into a larger network for the purpose of saving IP space and reducing routing table size

## Example

- Suppose Company A needs IP for 1000 machines
- Assign 4 *contiguous* Class C address blocks 192.60.128.0, 192.60.129.0, 192.60.130.0, 192.60.131.0 (last 8 bits 0)
- Supernet:
  - Address : 192.60.128.0
  - Netmask: 255.255.252.0 (last 10 bits 0)
- Also written as:
  - 192.60.128.0/22
  - 22 denotes size of network portion. Also called prefix.
  - Routing done by prefix




- Routing table at all higher level routers:
  - 192.60.128.0/22 - send to host X (next hop on way to Company A's router RA)
- Routing table at RA:
  - 192.60.128.0/24 – send to router of first net
  - 192.60.129.0/24 – send to router of second net
  - 192.60.130.0/24 – send to router of third net
  - 192.60.131.0/24 – send to router of fourth net
- Routers always do longest prefix match. If two entries match, longest match is taken.
  - Example: two entries in table: one for 192.65.0.0/16 and one for 192.65.128.0/24. If address is 192.65.128.4, second entry will be used even though it matches both.




## Advantage

- Routing table at higher levels will have only 1 entry for the 4 networks
- In classful addressing (that did not recognize masks), would have required 4 entries for the 4 networks
- Possible only due to contiguous allocation, so that higher level routers can just send it to lower level routers (in this case company A's router) using one entry only. Lower level router will distinguish.



- So routing table will contain networks (in prefix form or explicit net/mask form) and maybe some hosts (32 bit prefix), and a next hop address for each of them, plus some other information
- Usually a routing cache is also there. Cache contains recent routing decisions. Destination address first looked up in cache. If not found, longest-prefix match done in routing table.
- How is the routing table built and maintained?



## Static Routing

- Routing table updated manually (ex. *route* command in Linux)
- No dynamic update when network conditions change
- Fine for very small networks
- Not good for larger networks that interconnect larger no. of machines or networks



## Dynamic Routing

- Routers communicate among themselves to update routing tables dynamically
- No manual intervention needed normally
- Routing protocol goals
  - Optimal (compute best route)
  - Low overhead (should not send too many messages)
  - Robust (can handle unusual circumstances)
  - Rapid convergence when network changes
- Different routing protocols – RIP, OSPF, BGP, IS-IS...

*But first, we must understand the basic routing structure of the Internet...*



## Autonomous Systems

- Set of routers that use the same routing policy, has same administration,
- An AS may use the same routing protocol inside it, or more than one routing protocol
- Each AS has a unique number. AS numbers are allocated centrally just like IP addresses.
- From outside the AS, the AS is viewed as a single entity
- Public AS numbers are usually taken by service providers running their own routers



### • Interior Gateway Protocols (IGPs)

- Routing protocols within a single AS
- Ex. – RIP, OSPF, IS-IS
- Two different AS can use two different IGPs

### • Exterior Gateway Protocols (EGPs)

- Routing protocols between multiple AS
- Ex. – EGP, BGP



## Internet Routing

- Hierarchical router structures
- Routers within an AS run an IGP (ex. RIP/OSPF) to maintain their routing tables
- Designated routers from each AS run some EGP (ex. BGP)
- Allows better manageability as compared to one flat RIP or OSPF network

## RIP (Routing Information Protocol)

- RFC 1058 – RIP Version 1
- RFC 1388/2453 – RIP version 2 (1388 is obsolete, but good for starting)
- IGP based on Distance Vector method (Bellman-Ford algorithm for shortest path)
- Simple, efficient in small networks (so, limited to networks with max. 15 hop distance between nodes)
- Runs on UDP, port 520

## RIP Routing Table

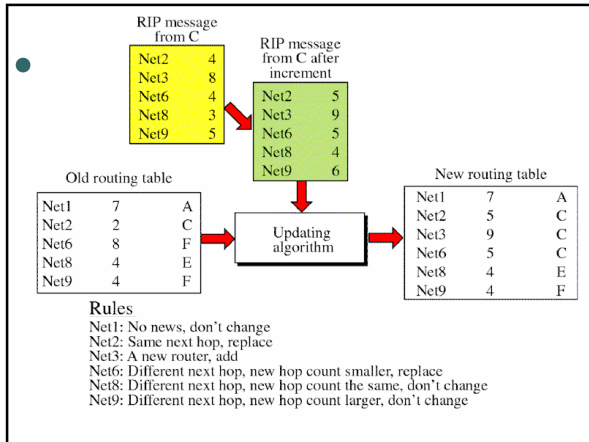
- Collection of routing entries
- Each routing entry contains
  - Host/network address
  - Next hop
  - Metric (no. of hops)
  - Route change flag (did the route change recently?)
  - ...
- Entries to reach all destinations within the system

## Routing Table Maintenance

- Each router initially knows only networks directly connected to it
- Each router sends an update message to all neighboring routers periodically (usually every 30 seconds)
- An update message will contain whole or part of the node's routing table (series of <network address + metric> entries)
- Receiving node checks for each route received in update message. Update an existing route if a better route is found.
- Update messages can also be sent in response to a specific request message

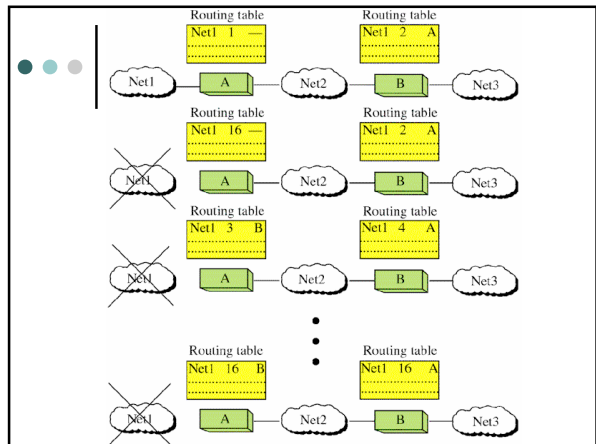
## Basic Update Algorithm

```
Update message received from router R
For each advertised destination in received message
  HC = advertised hop count + 1
  If (destination not in the routing table) /* New route */
    Add the advertised information to the table.
  Else
    If ( next-hop field in current route = R ) /* Why this? */
      Replace entry in the table with the advertised one
    Else If ( HC < hop count in the table ) /* better route */
      Add it to the routing table.
    Else do nothing
/* end for */
```



- ## Handling Topology Change
- Routes received from a router time out (usually after 180 seconds)
  - Timed out routes are set as invalid (metric set to 16 (infinity))
  - Route change flag is set to indicate this entry is changed
  - Routes deleted from table after some time (usually 120 sec. after the first timeout)

- ## Counting to Infinity Problem
- Routing Loops formed may take a long time to break
    - Nodes may have to update routes until the metric reaches an impossible value (*infinity*) to detect an invalid route
  - Lets see an example...



## RIP Solutions

- Limit hop count to 15 (16 = infinity)
- Split Horizon
  - In update message to R, do not include route advertisements that came from R in the first place
  - Solves routing loops involving two machines only
- Hold Down
  - If some route update says N is unreachable, don't modify N for some time even if a better route found
- Split Horizon with Poisoned Reverse
  - If route came from R, send route back with distance (metric) 16
- Triggered update
  - Send updated table immediately on a route change without waiting for the 30 second time period
- Counting to infinity still possible

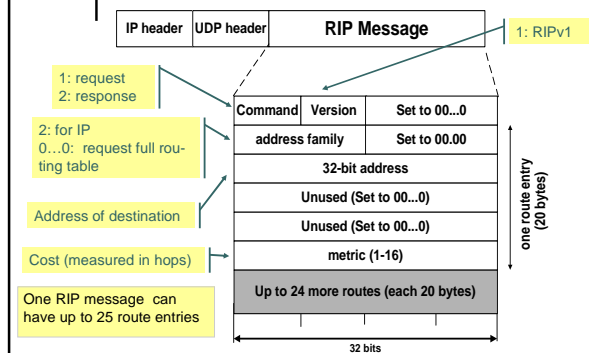
## Problems with RIP

- Slow convergence
  - If a routing entry changes, it may take a long time for all other routing tables to know this
  - Routing loops may take a long time to be detected (counting to infinity problem)
- Not suitable for very large networks
  - 15 hops maximum
- Support for only hop count as metric
- No support for subnets
  - Later extension adds limited support, still no support for CIDR or VLSM
- Periodic broadcasts may consume lot of bandwidth
- Only one best path kept, no scope for load balancing

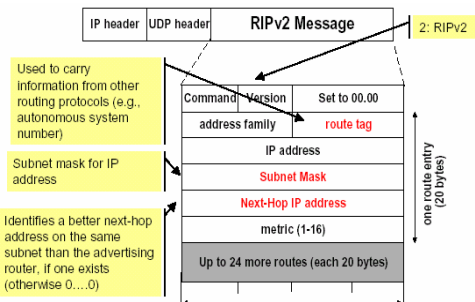
## RIP Version 2

- Supports CIDR and VLSM
- Supports simple authentication
- Allows sending of a *next hop* field in messages
- Still has the problems of metric, size, and slow convergence

## RIPv1 Packet Format



## RIPv2 Packet Format



- 4-byte header + max. 25 Route entries, each of 20 bytes
- Command
  - Request
  - Response (update – normal, in response to request, or triggered)
- Address Family
  - set to 2 for normal RIP packet
  - If set to 0xffff, means it is authentication route entry. Route Tag field will authentication type (only plaintext password now), next 16 bytes will contain password
    - So in this case, only 24 route entries can be there
- If more than 25 route entries in routing table, send additional RIP packets with the next route entries

## OSPF (Open Shortest Path First)

- Solves the problems with RIP + some
  - No hop count limit
  - Flexible metric
  - Scalable, allows for routing hierarchy within AS
  - Less b/w consumed
  - TOS (type-of-service) routing; allows specifying separate link metrics and separate routing tables for different TOS class
  - Better router authentication
  - External route tag
- Link-state protocol

## Link State Protocol

- Every router sends LSA (Link State Advertisement) containing each interface detail (name, connected to, metric) to all other nodes by flooding.
- Link State Update – collection of LSAs sent
- Link State database – collection of LSAs stored
- Every router collects all LSAs; gets a picture of the entire network; runs any shortest path algorithm (ex. Dijkstra's) to create shortest path tree. Routing table built from tree
- A new message can change the topology, changes routing table

## OSPF Basics

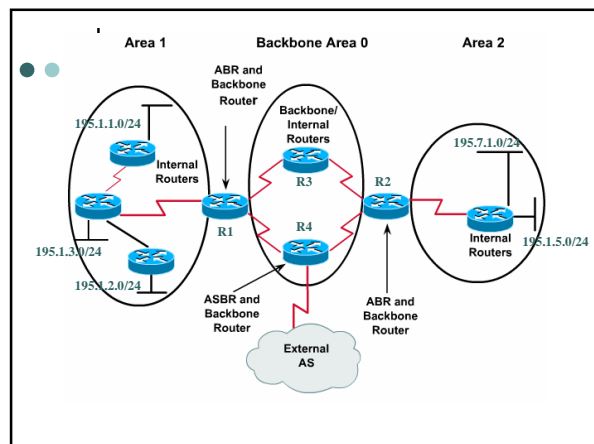
- RFC 2328 – OSPF Version 2
- Runs directly on IP (saves on UDP header length)
- LSA flooded initially from each router.
- As and when an LSA is received, part of the tree is built, routing table updated
- At steady state
  - all routers know the same network topology
  - Hello packets sent every 10 seconds (configurable) to neighbours
  - Absence of hello packet for 40 seconds indicate failure of neighbour – causes LSA to be flooded again
  - LSAs re-flooded every 30 minutes anyway

## Hierarchical Routing

- 2-level hierarchy
  - Routers divided into OSPF **areas**, each area has 32 bit id
  - All areas connected to special **OSPF backbone area** (area 0)
  - No routing information exchanged directly between areas; all exchanges through area 0
- Routing within area is by flooding LSAs as usual
- No LSA flooding between areas; networks within an area advertised to outside the area in summarized form

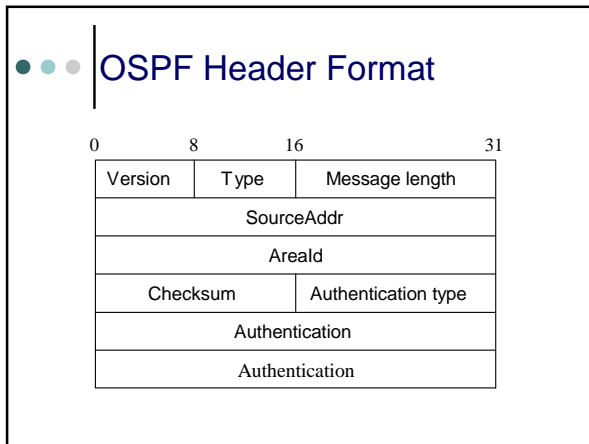
## Types of Routers

- **Internal** - Routers within an area (all interfaces connected within area)
- **Area Border Router** – Routers attached to 2 or more areas (at least one interface outside area)
  - Sends summarized distances to networks in one area to other areas
- **Backbone routers** – routes only within the backbone (at least one interface connected to area 0)
- **AS Boundary Routers** – connects to other ASs (at least one interface connected to another AS)



- Router R1 will advertise <195.1.0.0/16, cost> to backbone routers (if aggregation of networks done, cost is chosen to be maximum from R1 among the networks aggregated)
- Router R2 will advertise <195.7.1.0/24, cost1> and <195.1.5.0/24, cost2> (no aggregation of routes possible)
- Backbone routers just forward routes (adding cost)
- When R1 receives routes advertised by R2, it adds cost of final hop. If more than one route with same cost, one of them kept. Similar for R2
- So ABR's and Backbone routers are basically running distance vector protocol!!
- Loss of information, but better scalability

- ## Load Balancing
- If multiple routes to a destination with same cost (so different next hop), all routes kept
  - Upto the routers to use this
  - Example
    - Per destination: if two routes to same destination network, switch all traffic for host1 (in that network) on first route, host2 on second route
    - Per packet: send first packet on first route, second one on second...



- Packet types :
  - 1 : Hello (check if neighbor is up)
  - 2 : Database Description (sync. Database at beginning)
  - 3 : Link State Request (request specific LSA)
  - 4 : Link State Update (LSAs flooded)
  - 5 : Link State Acknowledgement (flooded LSAs are explicitly ack'ed – reliable flooding)
- Authentication type:
  - Cleartext
  - Encrypted (RFC details MD5 Hash, others possible)

## Link State Update Packet

- OSPF Packet Header
- 32-bit field containing number of LSAs in this packet
- List of LSAs follows. LSAs can be of different types. An example of a router-LSA is shown next

## Router-LSA

|                          |         |        |
|--------------------------|---------|--------|
| LS Age                   | Options | Type=1 |
| Link state ID            |         |        |
| Advertising router       |         |        |
| LS sequence number       |         |        |
| LS checksum              |         | Length |
| 0                        | Flags   | 0      |
| Number of links          |         |        |
| Link ID                  |         |        |
| Link data                |         |        |
| Link type                | Num_TOS | Metric |
| Optional TOS information |         |        |
| More links               |         |        |

### First 20 bytes is header

- LS Age – Age of the LSA. Incremented at routers as LSA is flooded, also as LSAs are held. Increment upto MaxAge (=1 hour default)
- Type – 1 = router-LSA. Four other types possible
- Link State ID – value depends on LS-Type
- LS Sequence Number – used to distinguish different instances of the LSA. Larger means more recent

### Rest is the LSA

- Link Type – specifies type of link (point-to-point etc.)
- Link ID/Link Data – depends on type. For type 1 (point-to-point), this will just be the IP addresses of the router at the other end)
- Num\_TOS – number of additional metrics specified, one for each TOS (0 if none specified)
- Metric – default metric. If TOS specified, optional TOS information will specify the TOS and the corresponding metric