

# CS314s2 - lectures 22-31

## How the Internet works

Dr Brian Carpenter

- 22, 23 – Routing
  - 24, 25 – IP version 4, DNS
  - 26 – IP version 6
  - 27 – Transport protocols: TCP
  - 28 – Transport protocols: UDP; Sockets
  - 29 – Applications (SSH, FTP, SMTP,...)
  - 30 – Applications (HTTP and the Web)
  - 31 – Summary
- Dates: 25, 27, 28 Sept., 2, 4, 5, 9, 11, 12, 16 Oct.
  - Approximately covering Shay chapters 10.4 to 12
  - **Questions:** [brian@cs.auckland.ac.nz](mailto:brian@cs.auckland.ac.nz) or room 303s.587 (most days between 10 a.m. and 4 p.m.)

1

## About the level of detail

- I tend to give more technical details than the text books, because I want all students to feel that they can see how the various protocols could be coded in a programming language.
  - I will also give some advanced references, which are for interest only and quite optional.
- The level of detail given in Shay or Halsall should be sufficient for the exam questions.

2

## CS314s2- 22/23

### Routing algorithms and protocols

- This is quite a hard topic.
- In two lectures, we can only scratch the surface to explain principles.

3

## Topics

- Routing – the problem
- Approaches
- Algorithms
  - Dijkstra
  - Bellman-Ford
- Protocols
  - RIP
  - OSPF
  - BGP
- Final points

4

## Routing – related to distance

- How do we connect devices

- In the same room?
- In the next room? (tens)
- In the next building? (100s)
- In the next department?
- In the next campus? (1000s)
- In the next country? (millions)
- In the next continent? (billions)

Cable/radio boundary

+ Admin boundary

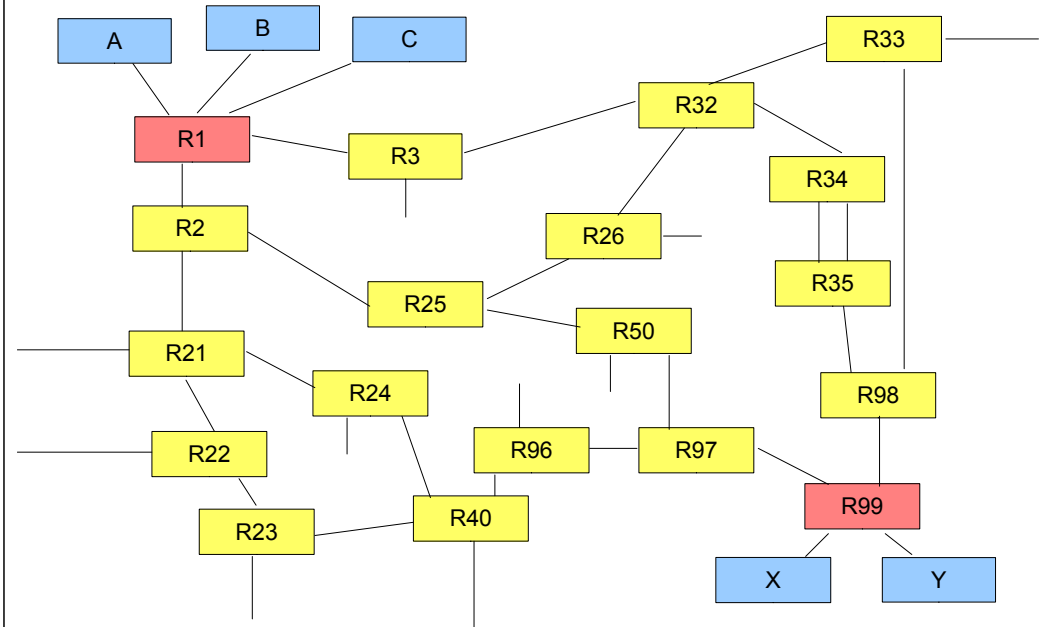
+ Funding boundary

+ Political boundary

Distance, logical separation, cost and numbers increase; must share costs and cables

5

## Routing – the problem (1)



## Terminology note

- This sort of picture shows what mathematicians call a *graph*, and there is a lot of *graph theory*.
- A graph consists of numbered *vertices* connected by *arcs*:  
 $V_1 \rightarrow V_2$  would be the arc from vertex 1 to 2
- For a program, a graph can be thought of as an array of vertices and a matrix of arcs
  - of course, not all arcs exist;  $R2 \rightarrow R3$  does not exist so its matrix entry will be zero.
- We will not dig deeply into graph theory here

7

## Graphs in programs

- Representing a graph:
  - Ordered list of vertex names:  
 $V_1 = 'A', V_2 = 'R1', \dots, V_{22} = 'R99', V_{23} = 'X', V_{24} = 'Y'$
  - List of arcs (i.e. a sparse matrix)  
 $A_i = (V_s, V_d, W)$ 
    - (source, destination, and a parameter for each arc)
- A path through the graph is represented by an ordered list of arcs
  - “Walking the graph” means an algorithm that starts at a given vertex and builds paths by searching the list of arcs that meet up (the destination of one arc is the source of the next).

8

## Routing – the problem (2)

- Choose the best path from A to X, knowing (at A) only the logical address of X.
- “Best” could mean
  - Smallest number of hops
  - Shortest time delay
  - Least congested
  - Cheapest
  - Administratively allowed
  - Easiest to discover
  - Any combination of the above
- Solution must be reasonably quick and guaranteed to avoid loops and deadlocks

9

## Routing – the problem (3)

- The real network is massive
  - Thousands of nodes on campus, millions per country, hundreds of millions worldwide.
- The real network is constantly changing
  - Nodes, routers and links added or removed constantly
  - View of network will be different in different places
- Therefore, algorithms must be scalable, dynamic and distributed
  - Central static route computation only works for small, very stable networks

10

## Approaches to dynamic routing

- Confine routing intelligence to specialised router boxes
  - End-systems mainly send everything to their default router
- Routers build connectivity maps and exchange routing information with their neighbours
- Two major approaches
  - Distance vector routing: router informs neighbours of its best paths. Each router calculates best paths for itself based on neighbour's paths.
  - Link state routing: router exchanges link status info with its neighbours. Each router relies on accuracy of its neighbours' status info in calculating best paths through the whole map.

11

## DV Routing: Bellman-Ford algorithm

- Router computes a table of its “distance” to each other router in the same network
  - Distance 1 = 1 hop, 2 = 2 hops etc.
- Sends this table to its neighbours
- Recomputes its own table using tables from its neighbours
- Works, but
  - Sending complete tables doesn't scale well and is slow to converge after a change
  - “Link down” condition can lead to loops, as the “distance” increases towards infinity

12

## Bellman-Ford: the computation (1)

- Compute the shortest path from R1 to all Rn knowing the topology of the router graph
  - Initialize each distance  $D(R1, Rn) = \infty$ , except  $D(R1, R1) = 0$
  - For each arc  $Rx \rightarrow Ry$  in the graph (starting with R1)
    - if  $D(R1, Ry) > D(R1, Rx) + 1$
    - then  $D(R1, Ry) := D(R1, Rx) + 1$
  - At this point, each  $D(R1, Rn)$  is the length of the shortest path from R1 to Rn
  - The algorithm also keeps track of the “next hop” router from R1 for the path

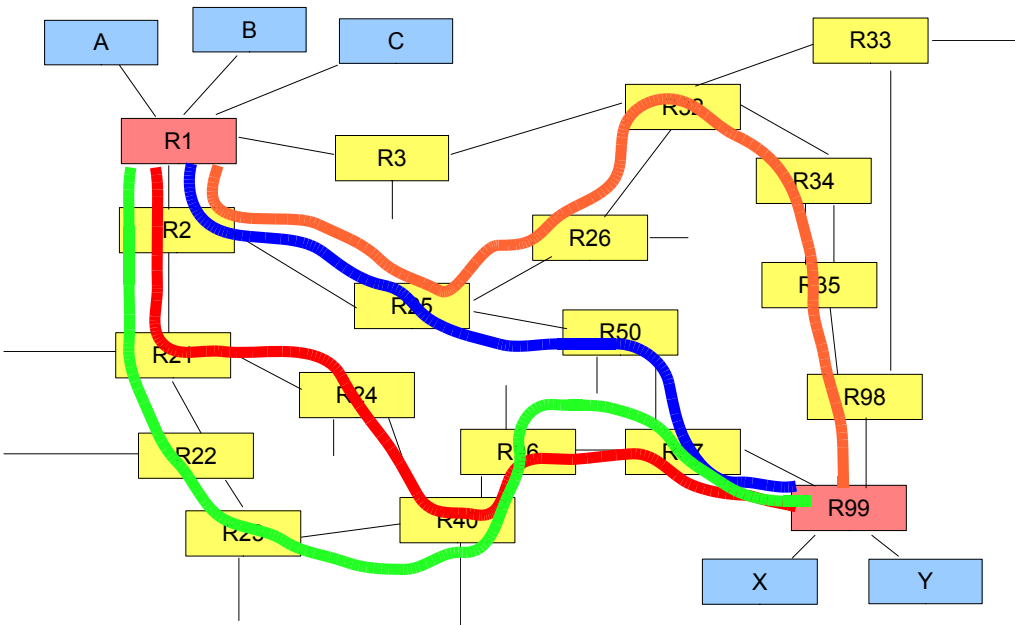
13

## Bellman-Ford: the computation (2)

- That dealt with paths from R1. For complete routing tables, run again starting from R2, R3...
- Note that algorithm processes every arc times every node!
- The “+1” above is a simplification where all paths have equal weight. It should be a weight  $W(x, y)$  to allow for administrative preferences for certain links:
  - if  $D(R1, Ry) > D(R1, Rx) + W(x, y)$
  - then  $D(R1, Ry) := D(R1, Rx) + W(x, y)$
- $W(x, y) < 0$  is allowed by the algorithm, but can lead to loops, which can be detected if any  $D(R1, Ry) > D(R1, Rx) + W(x, y)$  condition exists after the algorithm ends

14

## Bellman-Ford paths



## Link State Routing: Dijkstra algorithm

- Also called “shortest path first” or “SPF”
- First we'll describe the algorithm in the abstract
- Then a few comments

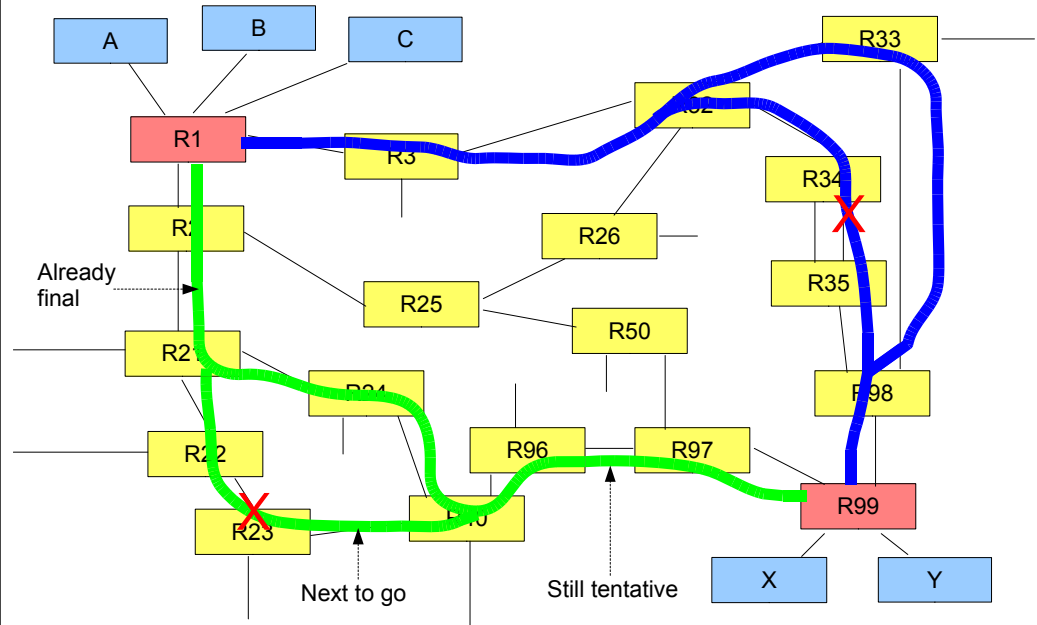
16

## Dijkstra: the computation

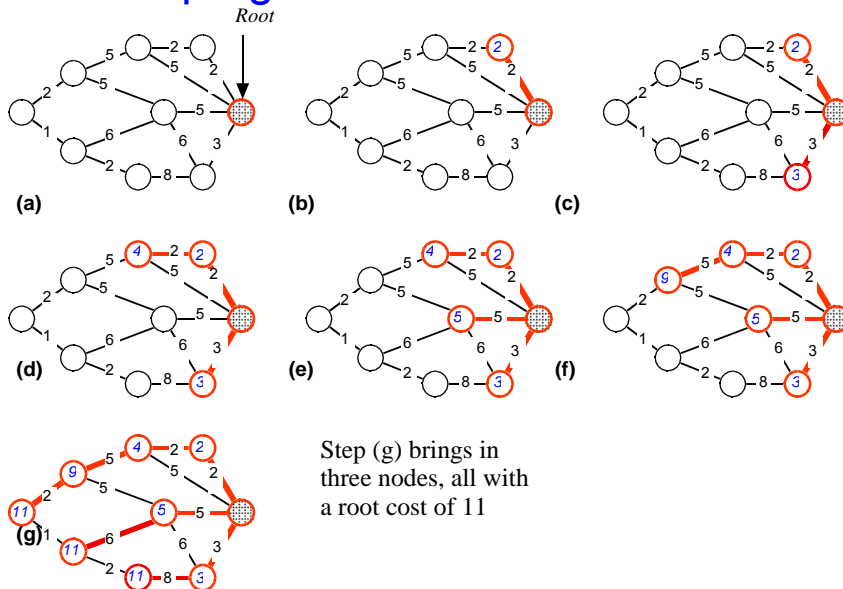
- Initialize each distance  $D(R1, Rn) = \infty$ , except  $D(R1, R1) = 0$
- Mark all D's as *tentative*, except  $D(R1, R1)$  as *final*
- For each arc  $Rx \rightarrow Ry$  in the graph (starting with R1)
  - if  $D(R1, Ry) > D(R1, Rx) + W(x, y)$
  - then  $D(R1, Ry) := D(R1, Rx) + W(x, y)$
- When all paths to  $Ry$  have been checked,  $Ry$  is removed from scanning of graph and its current  $D(R1, Ry)$  is marked as *final*
  - i.e. we only minimise the path to  $Ry$  *once*; in Bellman-Ford we did it for every path through  $Ry$
- $W(x, y) < 0$  is not allowed

17

## Dijkstra paths (snapshot)



## CS314 traditional Dijkstra example: progressive evaluation



## Link State: Comments

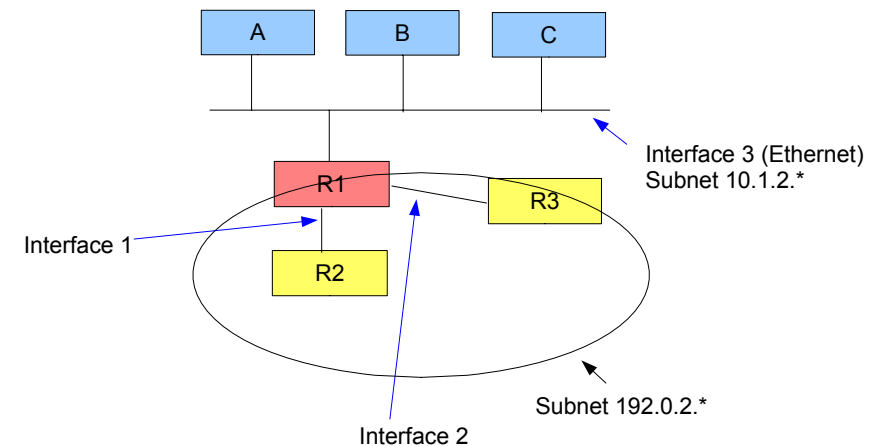
- Algorithm processes many fewer arcs than Bellman-Ford – once a distance is *final* that node is no longer considered
  - Whether this saving matters depends on size of graph and frequency of changes
- Also, each node receives neighbours' link state tables so doesn't have to start from  $D(R1, Rn) = \infty$ 
  - Trust neighbours' path lengths – OK if you can trust your neighbours
- Link State versus Distance Vector was quite a battle in the technical community some years ago, so we have a variety of routing protocols in service today. We'll talk about three: RIP (DV), OSPF (LS) and BGP (modified DV).

# RIP: Routing Information Protocol

- The original Internet DV protocol for on-site use
  - Remember that in real life, Bellman-Ford is executed as a distributed algorithm – each router computes the paths from itself to others.
  - Thus, each router keeps a table of destinations:
    - IP address of destination
    - IP address of first hop towards destination
    - Interface number for first hop
    - Distance metric for destination
    - Timestamp when entry was last updated
  - The router's complete distance vector can be read out of this table

21

## R1 and its interfaces



22

## Notional Initial DV table for R1

Name	Dest IP	1 <sup>st</sup> hop IP	Interface	Distance	Timestamp
R1	192.0.2.1	0.0.0.0	0	0	0
R2	192.0.2.2	0.0.0.0	1	∞	0
R3	192.0.2.3	0.0.0.0	2	∞	0
A	10.1.2.3	0.0.0.0	3	∞	0
B	10.1.2.5	0.0.0.0	3	∞	0
C	10.1.2.17	0.0.0.0	3	∞	0

### Notes:

- R1 points to itself using the loopback address and distance zero.
- The other distances are set to infinity.
- R1 also shows itself as the 1<sup>st</sup> hop to its neighbours.
- All timestamps are initialized before we start.
- In practice, end systems like A, B, C don't run RIP; the router can discover them on the LAN and is pre-configured with the summarised destination....

23

## Actual Initial DV table for R1

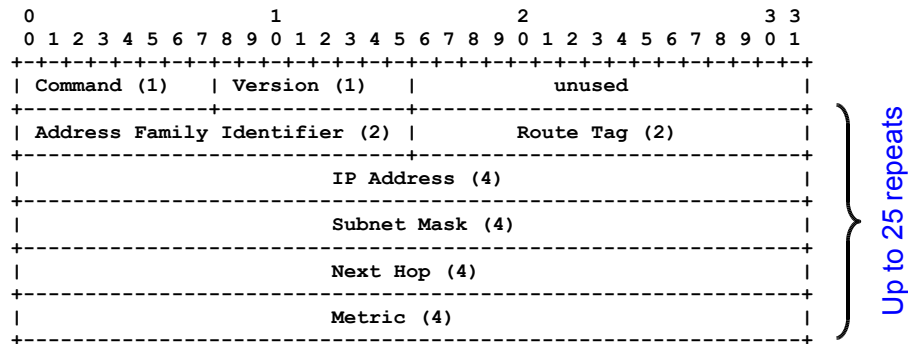
Name	Dest IP	1 <sup>st</sup> hop IP	Interface	Distance	Timestamp
R1	192.0.2.1	0.0.0.0	0	0	0
R2	192.0.2.2	0.0.0.0	1	16	0
R3	192.0.2.3	0.0.0.0	2	16	0
None	10.1.2.*	0.0.0.0	3	1	0

### Notes:

- There is no way out of configuring this externally; the router cannot guess any of it.
- In reality, destinations are address + bit mask; A, B and C can be summarised as address 10.1.2.0, mask 255.255.255.0 . This means all addresses from 10.1.2.0 to 10.1.2.255 .
- 10.1.2.\* stands for 10.1.2.0, mask 255.255.255.0 .
- RIP uses 16 as infinity
- Now we have a simple distance vector that can be passed to R2 and R3.

24

## RIP messages from router to router



Get used to this form of protocol data unit drawing – it's how all basic Internet standards show bits and bytes. This shows 24 bytes in 6 rows of 4 bytes.

25

## RIP protocol fields

- Command
  - 1: request sending of neighbour's DV
  - 2: response including sender's DV
- Version
  - Version of RIP in use – should be 2
- AFI
  - 2 for IPv4
- Route tag
  - A value that distinguishes routes within the RIP domain from routes that go outside the RIP domain (e.g. wide area routes)
- Address and mask – as above
- Next hop address – if not the current router
  - Zero means the default route via current router
- Metric
  - A value up to 15 (number of hops)

26

## Why is infinity equal to 16?

- By convention, RIP considers a metric of 16 to be infinity.
- Real networks should never be designed so that any normal path needs 16 hops
  - Even my messy example has at most 8 hops
- By having a low value for “infinity”, RIP will not waste too many cycles when re-computing routes after a link breakage
  - Even if a destination has become unreachable, we will stop looking for a path when the length exceeds 15

27

### Background slide

## Constructing a RIP packet for R1 (1)

```

Command: (2)      00000010
Version: (2)      00000010
Unused:           00000000 00000000
AFI,tag: (2+2)    00000000 00000010 00000000 00000000
R1's address:     11000000 00000000 00000010 00000001
Mask: (none)      00000000 00000000 00000000 00000000
Next hop:(self)   00000000 00000000 00000000 00000000
Metric: (0)       00000000 00000000 00000000 00000000
AFI,tag: (2+2)    00000000 00000010 00000000 00000000
R2's address:     11000000 00000000 00000010 00000010
Mask: (none)      00000000 00000000 00000000 00000000
Next hop:(self)   00000000 00000000 00000000 00000000
Metric: (16)      00000000 00000000 00000000 00010000
    
```

28

## Constructing a RIP packet for R1 (2)

```

AFI,tag: (2+2)  00000000 00000010 00000000 00000000
R3's address:   11000000 00000000 00000010 00000011
Mask: (none)    00000000 00000000 00000000 00000000
Next hop:(self) 00000000 00000000 00000000 00000000
Metric: (16)    00000000 00000000 00000000 00010000
AFI,tag: (2+2)  00000000 00000010 00000000 00000000
LAN address:    00001010 00000001 00000010 00000000
Mask:           11111111 11111111 11111111 00000000
Next hop:(self) 00000000 00000000 00000000 00000000
Metric: (1)     00000000 00000000 00000000 00000001

```

29

## How RIP operates

- Each router sends its full DV in one or more Response messages
  - On startup
  - Every 30 seconds
  - Whenever something changes (e.g. a link goes down)
  - When a Request message is received
- Recalculates its DV whenever it receives a Response
  - Which includes running Bellman-Ford algorithm
- That is enough for all routers to have up-to-date routing tables
  - Unless rate of changes is too great for the messages and calculations to keep up

30

## Comments on RIP

- We've skipped some details.
- Obviously, RIP messages are quite noisy with many unused bits.
- Convergence time can be large when conditions are bad (e.g. recovering from power failure in a building)
- RIP therefore doesn't work well in large, complex networks.
- Generally, OSPF is considered better these days.

31

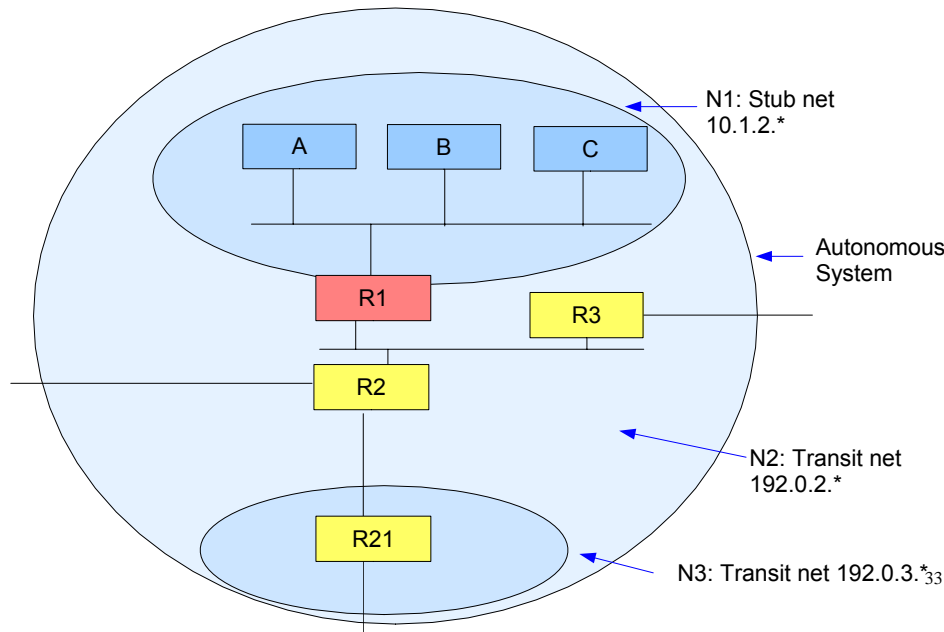
## OSPF: Open Shortest Path First

- A common Internet LS protocol for on-site use
  - Dijkstra is executed as a distributed algorithm – each router computes its own shortest paths.
  - Conceptually, OSPF runs within a finite domain known as an Autonomous System (AS).
    - It distinguishes routers from networks within the AS.
    - Networks can be stub or transit networks.
  - Each router keeps a Link State database,
  - The router's Link State Advertisements (LSAs) to its neighbours can be read out of this database.

32



## OSPF conceptual model



## Contents of Link State database

- The vertices in the network graph will be:  
**N1, N2, N3, R1, R2, R3, R21**
- Arcs will be  **$R_x \rightarrow R_y$ ,  $N_x \rightarrow R_y$  or  $R_x \rightarrow N_y$**  (never  **$N_x \rightarrow N_y$** )
- For each destination, LS database lists
  - IP address (and mask) of destination
  - IP address of first hop towards destination
  - Interface number for first hop
  - Distance metric for destination
- External networks (outside the AS) will typically have large metrics

34

## Splitting very large AS's

- OSPF already allows an AS to contain logically separate networks.
- For a large campus, it is possible to split the AS into logically separate “areas” which each run their own SPF computations.
  - Reduces overhead within each area
  - Area 0 then acts as a campus backbone
  - Inter-area routes cross the backbone
- More efficient operation in a large network
  - Glitches in one department don't affect another

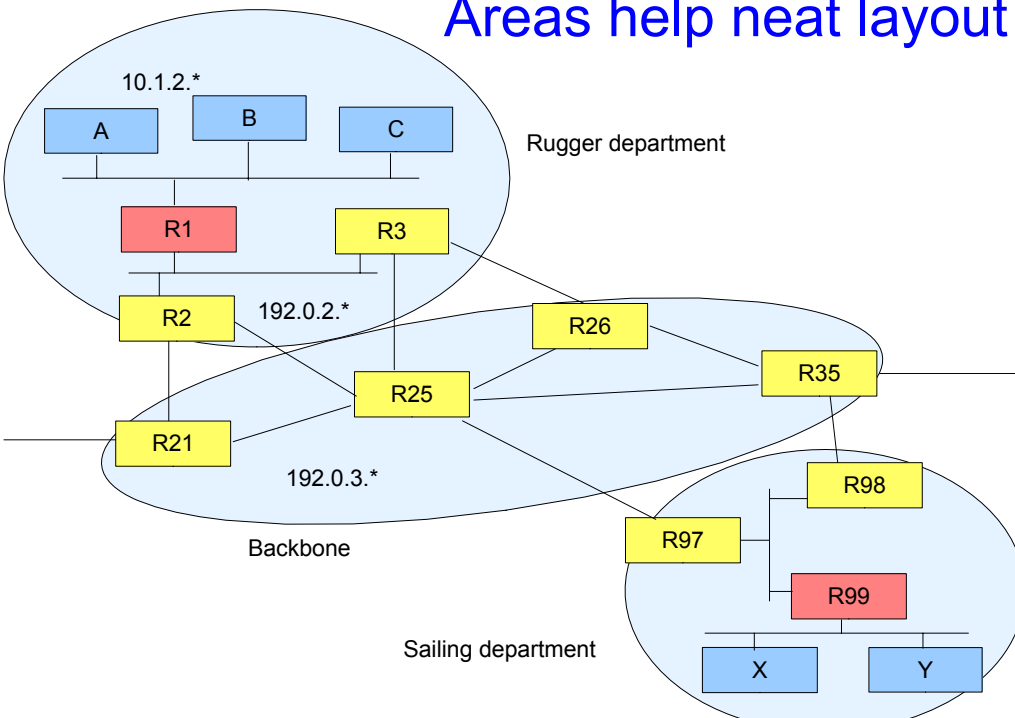
35

## Flavours of OSPF router

- **Internal**
  - Connected to LANs and links within an Area
- **Designated**
  - When >1 routers on a LAN, one is designated to send LSAs
- **Area Border**
  - Connected to the backbone Area and at least one normal Area. Computes routes for each.
- **Backbone**
  - Connected to the backbone Area
- **AS Boundary**
  - Connected to the outside world
  - Advertises routes to external networks

36

## Areas help neat layout



## OSPF message types

- 1 Hello Discover/maintain neighbours
- 2 Database Description Summarise database contents
- 3 Link State Request Request database download
- 4 Link State Update Send database update
- 5 Link State Ack Acknowledge an update  
(not abbreviated as LSA)

- Link State Updates carry the LSAs (Link State Advertisements) of the sending router and LSAs from its upstream routers.
- Within an Area, LSAs describe link state in detail, similar to RIP
- Beyond an Area, LSAs summarise routes per network <sup>38</sup>

## What causes an LSA to be updated?

- Its lifetime expires
- Interface up/down
- Designated router for a LAN changes
- Change of state of neighbour router
- Intra-area route changes in border router
- Inter-area route changes
- Border router attaches to new area
- Backbone topology change (virtual route change)
- External route change
- AS Boundary router down

## What's in an OSPF LSA

- Five types of LSA:
  - 1 Router-LSAs – link state for each interface
  - 2 Network-LSAs – routers for a transit subnet
  - 3,4 Summary-LSAs – inter-Area link state
  - 5 AS-external-LSAs – external link state

## Main items in a Router-LSA

```

LS type = 1                ;indicates router-LSA Link State
Link State ID = 192.0.2.1   ;R1's Router ID
Advertising Router = 192.0.2.1 ;R1's Router ID

#links = 2

    Link ID = 192.0.2.3     ;IP address of Desig. Rtr.
    Link Data = 192.0.2.1   ;R1's IP interface to net
    Type = 2                ;connects to transit net
    metric = 1

    Link ID = 10.1.2.0      ;IP Network number
    Link Data = 255.255.255.0 ;Network mask
    Type = 3                ;connects to stub net
    metric = 2
  
```

41

## Main items in a Network-LSA

```

LS type = 2                ;indicates network-LSA Link State
Link State ID = 192.0.2.3   ;IP address of Desig. Rtr.
Advertising Router = 192.0.2.3 ;R3's Router ID

Network Mask = 255.255.255.0

    Attached Router = 192.0.2.3 ;Router ID
    Attached Router = 192.0.2.1 ;Router ID
    Attached Router = 192.0.2.2 ;Router ID
  
```

42

## How OSPF operates

- Each router sends its LSAs in one or more Link State Update messages
  - When timer expires
  - Whenever an LSA changes
  - When a LS Request message is received
- Recalculates its LSAs whenever it receives an LSU Response
  - Which includes running Dijkstra algorithm
- That is enough for all routers to have up-to-date routing tables
- We've skipped many details.
  - Convergence times and scalability significantly better than RIP

43

## BGP4: Border Gateway Protocol v4

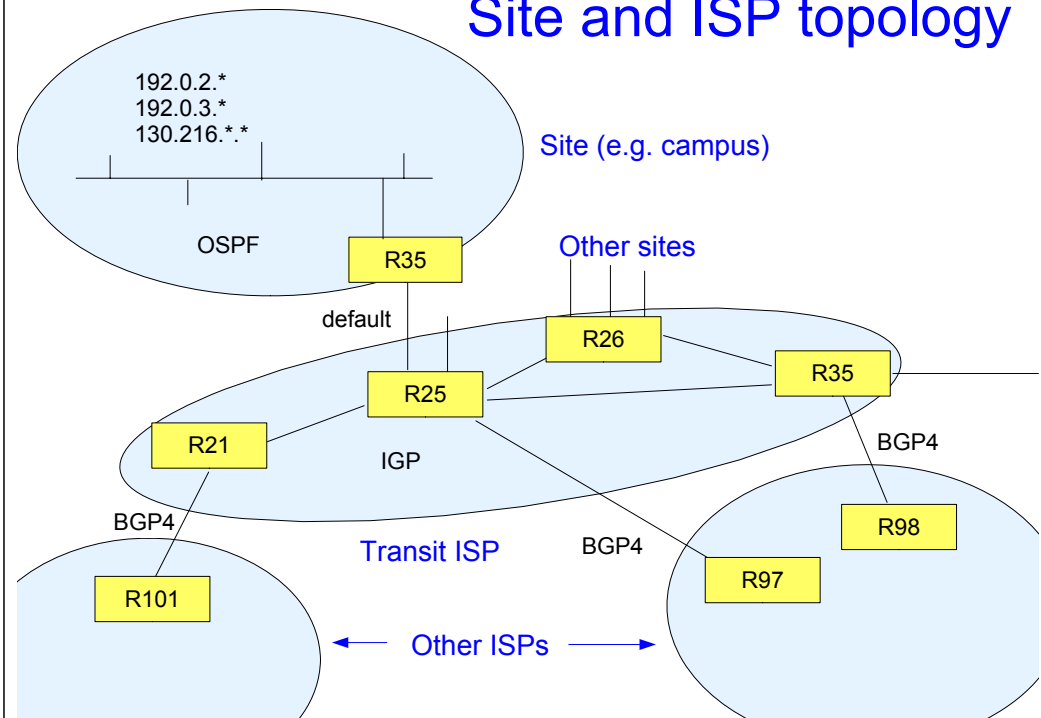
- The Internet distinguishes two main classes of routing protocols
  - Interior Gateway Protocols (IGPs) used within an organisation or campus, e.g., RIP or OSPF
  - Exterior Gateway Protocols (EGPs) used between organisations or campuses, e.g. BGP4
    - BGP4 has no competition today.
    - EGPs are also sometimes called Inter-Domain Routing (IDR) protocols
- BGP4 is a modified Distance Vector protocol, sometimes called a Path Vector protocol

44

## The problem BGP solves

- Assume every site is running an IGP (OSPF or an alternative). The world now needs to find routes to the site, for a set of networks, e.g. 192.0.2.\*, 192.0.3.\*, 130.216.\*.\*
  - A site may have several unrelated address blocks.
  - A consumer ISP looks like a site
  - In the world, there are hundreds of 1000s of sites
- How can any site find routes to any other site?
  - Starting point: the *default route* 0.0.0.0 in the IGP points to a site exit router
  - The ISP accepts packets via an ingress router
  - It needs IGP routes to its customer sites and BGP routes to everything else

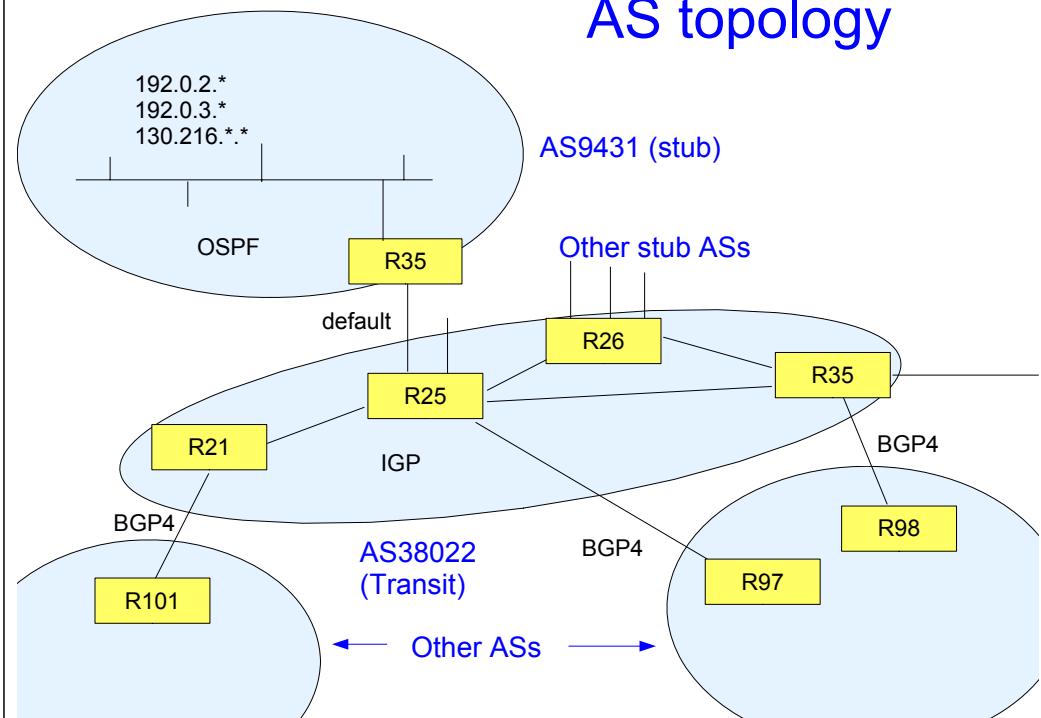
## Site and ISP topology



## BGP4 AS Paths

- BGP formalises the concept of Autonomous System
  - Over-simplifying, each ISP and each major customer has an AS number.
  - Several ASs can in fact be federated so they appear externally as a single AS.
- An AS path is basically a list of AS numbers that leads to a given destination
  - A destination is a network address/mask, usually called a *prefix* in BGP
  - 130.216/16 means a 16 bit prefix: 130.216.0.0 with mask 255.255.0.0

## AS topology



## About AS numbers

- An AS path to 130.216/16 will end with ..., AS38022, AS9431
- AS numbers are currently 16 bits but will be expanded to 32 bits, since >43,000 have been handed out already.

49

## BGP messages

- Open
  - Sent when a link to a neighbour BGP speaker comes up
- Update
  - Routing information (see below)
- Notification
  - Sent when a link to a neighbour BGP speaker is intentionally closed
- Keepalive

*Note* – no Request message in BGP

50

## BGP update messages

- BGP speakers send updates to their BGP neighbours
- They contain various *attributes* per destination
  - Origin of information (IGP or EGP)
  - AS Path
  - IP address of next hop (if not the sender of the update)
  - Unreachable (if announcing a path failure)
  - Metric (only within an AS; BGP does not weight links)
  - Community (see below)

51

## BGP communities

- BGP supports routing *policies* instead of weights.
- The *community* attribute, if used, acts as a label for a group of destinations, so that policy rules can be applied to the group.
  - Artificial example rule: do not use paths to this community that include AS5168.

52

## BGP operation

- When a BGP router receives an update
  - It filters the update according to local policy rules, discarding AS paths that its policy rejects.
  - Then runs Bellman-Ford over the resulting DV table (but uses AS path length as the metric, hence the phrase “path vector”).
  - Then generates update messages for its other neighbours.
- When a BGP router observes a link failure
  - It updates its DV table, runs Bellman-Ford, and generates update messages for its neighbours.

53

## Aggregation

- Today's global BGP4 table has about 230,000 entries (>5 per AS number).
  - See history at <http://bgp.potaroo.net/>
- It's only that small because BGP *aggregates* adjacent prefixes in a binary tree.
  - 192.0.2.0/24 and 192.0.3.0/24 can be aggregated as 192.0.2.0/23, etc.
  - Two prefixes that share an AS path are aggregated if possible, before advertising the path.

54

## BGP in practice

- BGP4 is the routing protocol that makes the Internet work.
  - Without BGP aggregation, the Internet would have melted down many years ago, with several times more entries in the global table.
  - Today, hardware improvements are roughly keeping up with growth.
  - But the number of dynamic updates is some cause for concern. What if updates start to arrive faster than they can be processed and passed on to the neighbours?

55

## Missing topics: MPLS and IS-IS

- No time in this course to discuss MPLS (MultiProtocol Label Switching).
  - This is a “layer 2.5” technique for building semi-permanent switched paths across a backbone network.
  - Used today by many ISPs underneath the traditional IGP/EGP router layer.
- We also haven't discussed IS-IS (Intermediate System - Intermediate System).
  - An alternative to OSPF used by many ISPs.

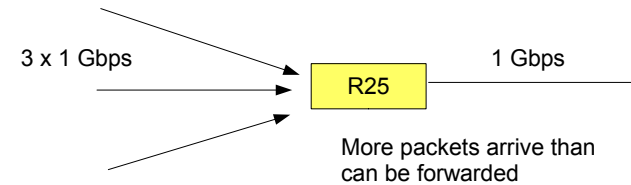
56

## RIBs and FIBs – building a router

- RIB: Routing Information dataBase
  - The routing table resulting from RIP, OSPF, BGP4, etc.
  - Generally, it's computed and accessed by software.
- FIB: Forwarding Information dataBase
  - A simple table that tells the hardware which interface to send a packet on:
  - IP Address – Mask – Interface number
  - Generally written by software, and read by specialist hardware operating at line speed (100 Mbps to several Gbps)
  - At 1 Gbps you have ~4 microseconds to process a packet, so the FIB lookup must take a fraction of a microsecond.

57

## Handling congestion



- Normally, networks are designed to avoid such bottlenecks, but sometimes bursts of traffic will cause congestion and buffer overflow.
  - Buffer overflow at both ends of a link could cause deadlock, with both ends unable to send to the other.
- IP is an “unreliable” protocol - congested routers discard excess traffic unpredictably when buffers fill up.
  - We'll discuss later how this affects upper layer protocols

58

## Routing – related to distance

- How do we connect devices
 

– In the same room?	Hubs, switches
– In the next room? (tens)	Cable/radio boundary
– In the next building? (100s)	OSPF + Admin boundary
– In the next department?	+ Funding boundary
– In the next campus? (1000s)	BGP + Political boundary
– In the next country? (millions)	MPLS
– In the next continent? (billions)	Distance, logical separation, cost and numbers increase; must share costs and cables

59

## References

- Shay 10.3 to 10.8
- Advanced text book (*completely optional*):

Interconnections (2<sup>nd</sup> edition)  
by Radia Perlman (Addison-Wesley)  
Published 1999 but still unbeaten

- The technical specs for all Internet protocols are published free as numbered “RFCs” (originally “Request for Comment”). They make tough reading and are *completely optional* for this course.  
<http://www.rfc-editor.org/rfcsearch.html>
  - RIP: RFC 1723
  - OSPF: RFC 2328
  - BGP4: RFC 4271

60