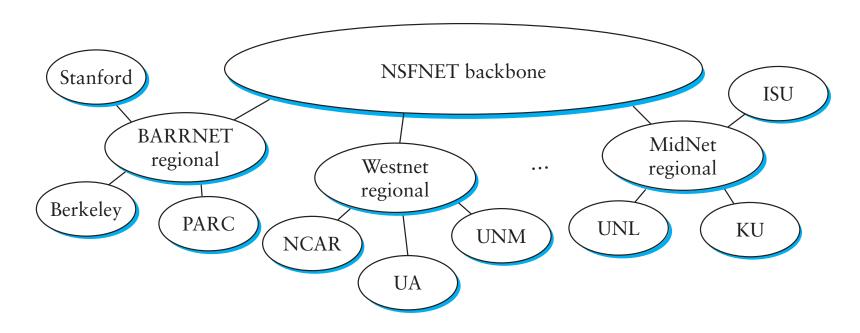
Lecture 6: Intra-Domain Routing

Overview

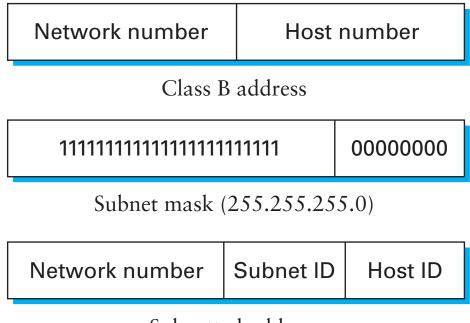
- Internet structure, ASes
- Forwarding vs. Routing
- Distance vector and link state
- Example distance vector: RIP
- Example link state: OSPF

The Internet, 1990



• Hierarchical structure w. single backbone

Address allocation, 1990



Subnetted address

• Hierarchical IP addresses

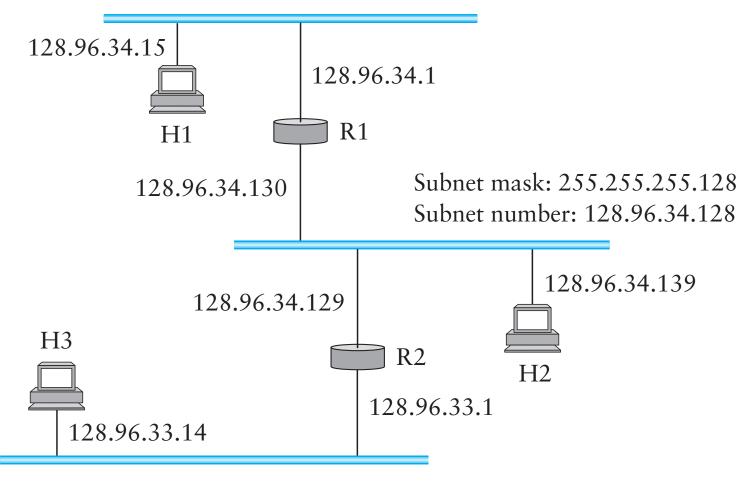
- Class A (8-bit prefix), B (16-bit), C (24-bit)

• Subnetting adds another level within organizations

- Subnet masks define variable partition of host part
- Subnets visible only within site

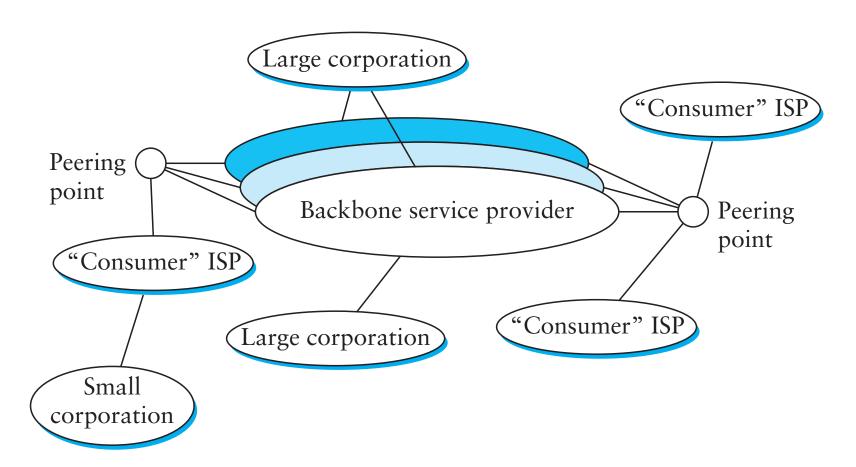
Example

Subnet mask: 255.255.255.128 Subnet number: 128.96.34.0



Subnet mask: 255.255.255.0 Subnet number: 128.96.33.0

The Internet, today



• Multiple "backbones"

Address allocation, today

• Class system makes inefficient use of addresses

- class C with 2 hosts (2/255 = 0.78%) efficient
- class B with 256 hosts (256/65535 = 0.39%) efficient)
- Causes shortage of IP addresses (esp. class B)
- Makes address authorities reluctant to give out class Bs

• Still Too Many Networks

- routing tables do not scale
- route propagation protocols do not scale

Supernetting

- Assign block of contiguous network numbers to nearby networks
- Called CIDR: Classless Inter-Domain Routing
- Represent blocks with a single pair (first network address, count)
- Restrict block sizes to powers of 2
 - Represent length of network in bits w. slash
 - E.g.: 128.96.34.0/25 means netmask has 25 1 bits, followed by 7 0 bits, or 0xffffff80 = 255.255.255.128
 - E.g.: 128.96.33.0/24 means netmask 255.255.255.0
- All routers must understand CIDR addressing

IP Connectivity

- For each destination address, must either:
 - 1. Have prefix mapped to next hop in forwarding table, or
 - 2. know "smarter router"—default for unknown prefixes
- Route using longest prefix match, default is prefix 0.0.0.0/0
- Core routers know everything—no default
- Manage using notion of Autonomous System (AS)
- Two-level route propagation hierarchy
 - interior gateway protocol (each AS selects its own)
 - exterior gateway protocol (Internet-wide standard)

Autonomous systems

• Correspond to an administrative domain

- Internet is not a single network
- ASes reflect organization of the Internet
- E.g., Stanford, large company, etc.

• Goals:

- ASes want to choose their own local routing algorithm
- ASes want to set policies about non-local routing

• Each AS assigned unique 16-bit number

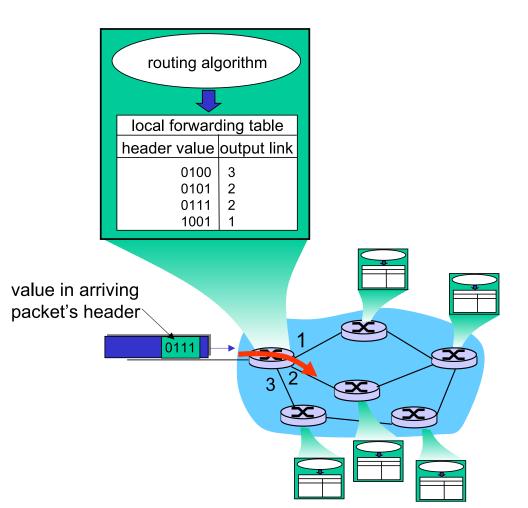
Types of traffic & AS

- Local traffic packets with src or dst in local AS
- Transit traffic passes through an AS
- Stub AS
 - Connects to only a single other AS
- Multihomed AS
 - Connects to multiple ASes
 - Carries no transit traffic
- Transit AS
 - Connects to multiple ASes and carries transit traffic

Intra-domain routing

- Intra-domain routing: within an AS
- Single administrative control: optimality is important
 - Contrast with inter-AS routing, where policy dominates
 - Next lecture will cover inter-domain routing (BGP)

What is routing?



• forwarding – moving packets between ports

- Look up destination address in forwarding table
- Find *out-port* or *\langle out-port*, *MAC addr\rangle* pair

• Routing is process of populating forwarding table

- Routers exchange messages about nets they can reach
- Goal: Find optimal route for every destination
- ... or maybe good route, or just any route (depending on scale)

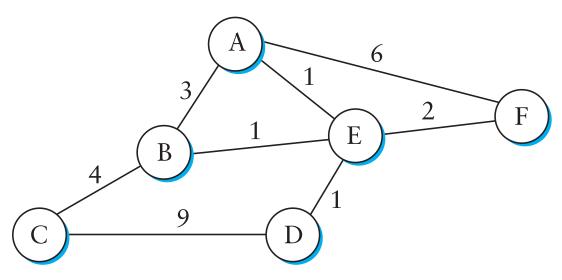
Stability

- Stable routes are often preferred over rapidly changing ones
- Reason 1: management
 - Hard to debug a problem if it's transient
- Reason 2: higher layer optimizations
 - E.g., TCP RTT estimation
 - Imagine alternating over 500ms and 50ms routes
- Tension between optimality and stability

Routing algorithm properties

- Global vs. decentralized
 - Global: All routers have complete topology
 - Decentralized: Only know neighbors & what they tell you
- Intra-domain vs. Inter-domain routing
 - Intra-: All routers under same administrative control
 - Intra-: Scale to \sim 100 networks (e.g., campus like Stanford)
 - Inter-: Decentralized, scale to Internet
- Today's lecture covers basic algorithms for intra-domain routing and two protocols: RIP and OSPF

Optimality



- View network as a graph
- Assign *cost* to each edge
 - Can be based on latency, b/w, utilization, queue length, ...
- Problem: Find lowest cost path between two nodes
 - Each node individually computes the cost (some recent research argues against doing this, more on this later)

Scaling issues

- Every router must be able to forward based on any destination IP address
 - Given address, it needs to know "next hop" (table)
 - Naïve: Have an entry for each address
 - There would be 10^8 entries!
- Solution: Entry covers range of addresses
 - Can't do this if addresses are assigned randomly! (e.g., Ethernet addresses)
 - This is why *address aggregation* is important
 - Addresses allocation should be based on network structure
- What is structure of the Internet?

Basic Algorithms

• Two classes of intra-domain routing algorithms

• Link state

- Have a global view of the network
- Simpler to debug
- Require global state

• Distance vector

- Require only local state (less overhead, smaller footprint)
- Harder to debug
- Can suffer from loops

Link State

Strategy

- Send to all nodes (not just neighbors)
- Send only information about directly connected links (not entire routing table)

• Link State Packet (LSP)

- ID of the node that created the LSP
- Cost of link to each directly connected neighbor
- Sequence number (SEQNO)
- Time-to-live (TTL) for this packet

Reliable flooding

- Store most recent LSP from each node
- Forward LSP to all nodes but one that sent it
- Generate new LSP periodically
 - Increment SEQNO
- Start SEQNO at 0 when reboot
 - If you hear your own packet w. SEQNO= n, set your next SEQNO to n+1
- Decrement TTL of each stored LSP
 - discard when TTL = 0

Calculating best path

• Dijkstra's shortest path algorithm

• Let:

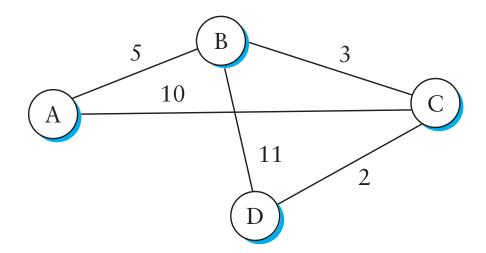
- *N* denote set of nodes in the graph
- l(i, j) denotes non-negative cost (weight) for edge (i, j)
- s denotes yourself (node computing paths)

• Initialize variables

- $M \leftarrow \{s\}$ (set of nodes "incorporated" so far)
- $C_n \leftarrow l(s, n)$ (cost of the path from s to n)
- $R_n \leftarrow \bot$ (next hop on path to n)

Dijkstra's algorithm

- While $N \neq M$
 - Let $w \in (N M)$ be node with lowest C_w
 - $M \leftarrow M \cup \{w\}$
 - Foreach $n \in (N-M)$, if $C_w + l(w,n) < C_n$ then $C_n \leftarrow C_w + l(w,n)$, $R_n \leftarrow w$
- Example: **D** $(D, 0, \perp)(C, 2, C)(B, 5, C)(A, 10, C)$



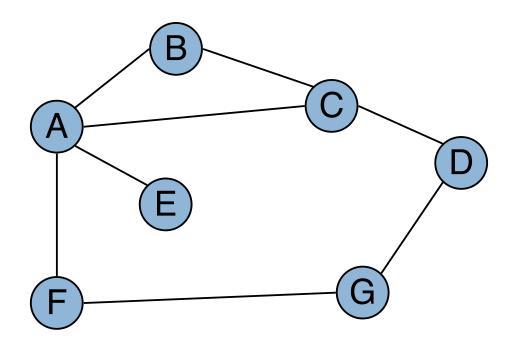
Distance Vector

- Local routing algorithm
- Each node maintains a set of triples
 - (Destination, Cost, NextHop)
- Exchange updates with direct neighbors
 - periodically (on the order of several seconds to minutes)
 - whenever table changes (called triggered update)
- Each update is a list of pairs:
 - (Destination, Cost)
- Update local table if receive a "better" route
 - smaller cost
 - from newly connected/available neighbor
- Refresh existing routes; delete if they time out

Calculating best path

- Bellman-Ford equation
- Let:
 - $D_a(b)$ denote the distance from a to b
 - c(a, b) denote the cost of a link from a to b
- Then $D_x(y) = min_z(c(x,z) + D_z(y))$
- Routing messages contain *D*; when does a node transmit a message?

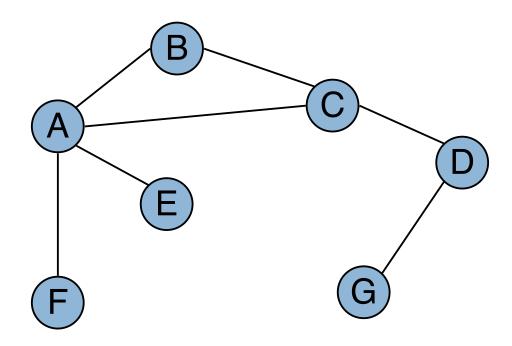
DV Example



$D/_{-}$		4-1-1	۱ ـ ـ ا
D S	routing	tabl	le:

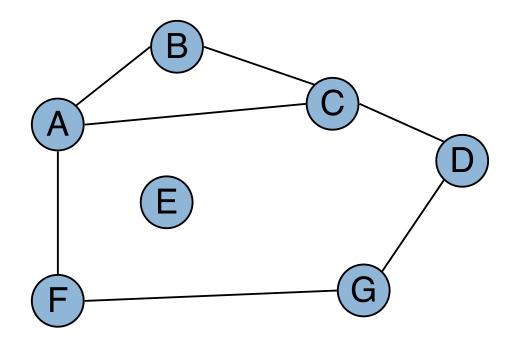
Cost	Nextriop	
1	A	
1	C	
2	С	
2	A	
2	A	
3	A	
	1 1 2 2 2	

Adapting to failures



- F detects that link to G has failed
- F sets distance to G to infinity and sends update to A
- A sets distance to G to infinity since it uses F to reach G
- A receives periodic update from C with 2-hop path to G
- A sets distance to G to 3 and sends update to F
- F decides it can reach G in 4 hops via A

Danger: Loops



- link from A to E fails
- A advertises distance of infinity to E
- B and C advertise a distance of 2 to E
- B decides it can reach E in 3 hops; advertises this to A
- A decides it can reach E in 4 hops; advertises this to C
- C decides that it can reach E in 5 hops...

How to avoid loops

- IP TTL field prevents a packet from living forever
 - Does not break or repair a loop
- Simple approach: consider small cost n (e.g., 16) to be infinity
 - After *n* rounds will decide node is unavailable
 - But rounds can be long: this takes time
- Distance vector based only on local information

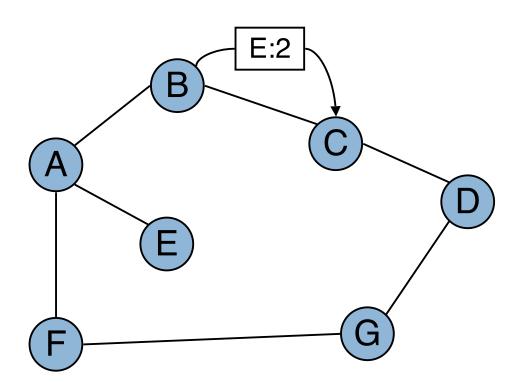
Better loop avoidance

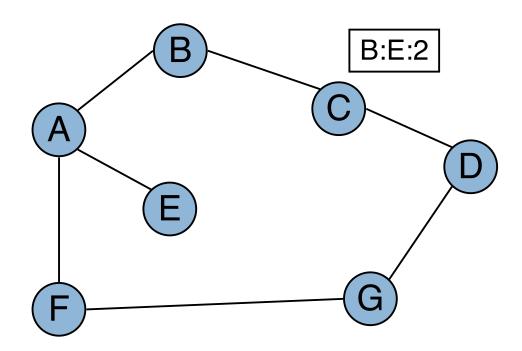
• Split horizon

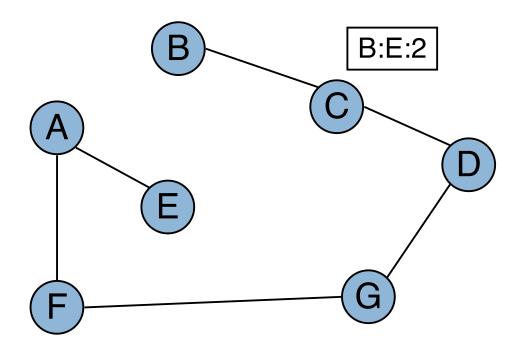
- When sending updates to node *A*, don't include destinations you route to through *A*
- Prevents *B* and *C* from sending cost 2 to *A*

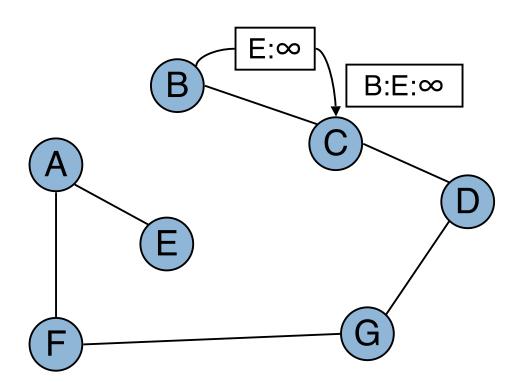
• Split horizon with poison reverse

- When sending updates to node A, explictly include very high cost ("poison") for destinations you route to through A
- When does poison reverse help?









Warning

- Note: Split horizon/split horizon with poison reverse only help between two nodes
 - Can still get loop with three nodes involved
 - Might need to delay advertising routes after changes, but will affect convergence time

Distance Vector vs. Link State

• # of messages

- DV: convergence time varies, but $\Omega(d)$ where d is # of neighbors of node
- LS: $O(n \cdot d)$ for n nodes in system

Computation

- DV: Could count all the way to ∞ if loop
- LS: $O(n^2)$

• Robustness – what happens with malfunctioning router?

- DV: Node can advertise incorrect path cost
- DV: Costs used by others, errors propagate through net
- LS: Node can advertise incorrect *link* cost

Metrics

• Original ARPANET metric

- measures number of packets enqueued on each link
- took neither latency nor bandwidth into consideration

New ARPANET metric

- stamp each incoming packet with its arrival time (AT)
- record departure time (DT)
- when link-level ACK arrives, compute Delay = (DT AT) + Transmit + Latency
- if timeout, reset DT to departure time for retransmission
- link cost = average delay over some time period

• Fine Tuning

- compressed dynamic range
- replaced *Delay* with link utilization

Today: policy often trumps performance [more later]

Intradomain routing protocols

- RIP (routing information protocol)
 - Fairly simple implementation of DV
 - RFC 2453 (38 pages)
- OSPF (open shortest path first)
 - More complex link-state protocol
 - Adds notion of *areas* for scalability
 - RFC 2328

RIPv2

- Runs over UDP port 520
- Limits networks to 15 hops (16 = ∞)
- Depends on count to infinity for loops
- Supports split horizon, poison reverse
- RFC 1812 specifes what options routers should or must have

RIPv2 packet format

RIPv2 Entry

0 1	2 3					
0 1 2 3 4 5 6 7 8 9 0 1	2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1					
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-	-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-					
address family identia	fier (2) Route Tag (2)					
+						
IP address (4)						
+						
Subnet Mask (4)						
+						
Next Hop (4)						
+						
Metric (4)						
+						

Route Tag Field

- Allows RIP nodes to distinguish internal and external routes
- E.g., encode AS

Next Hop Field

- Allows one router to advertise routes for multiple routers on same subnet
- Suppose only XR1 talks RIP2:

OSPFv2

- Link state protocol
- RFC 2453 (244 pages)
- Runs directly over IP (protocol 89)
 - Has to provide its own reliability
- All exchanges are authenticated
- Adds notion of areas for scalability

OSPF Areas

- Area 0 is "backbone" area (includes all boundary routers)
- Traffic between two areas must always go through area 0
- Only need to know how to route exactly within area
- Else, just route to appropriate area
- (Virtual links can allow distant routers to be in area 0)

OSPF areas

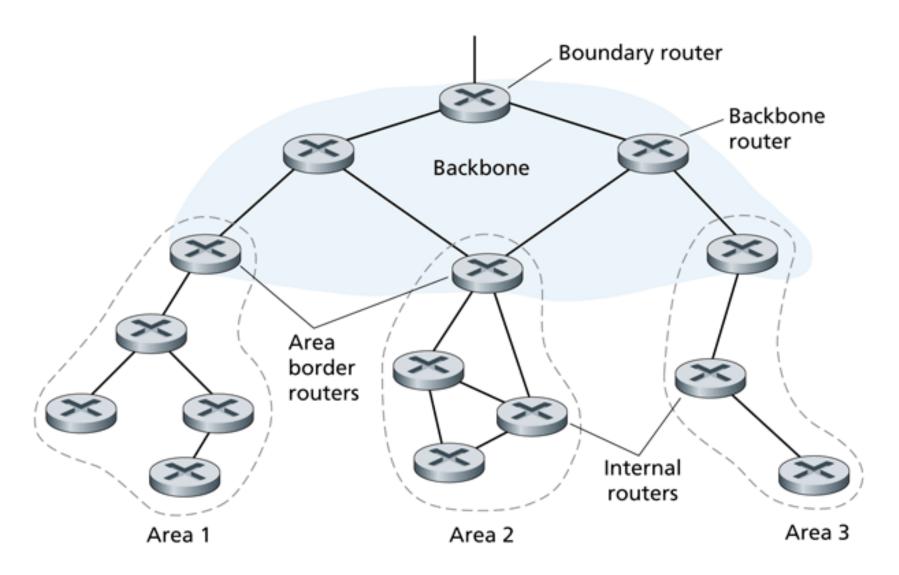


Figure 4.40 • Hierarchically structured OSPF AS with four areas

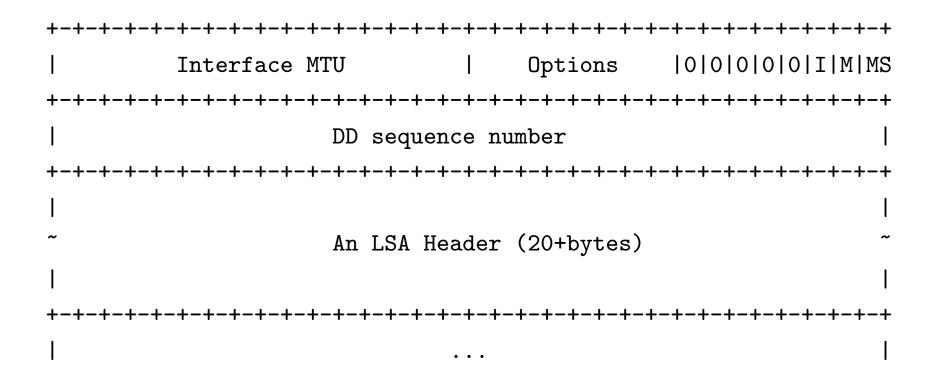
OSPF Packet Header

0	1	2		3		
0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5	6 7 8 9 0 1	2 3 4 5 6 7	8 9 0 1		
+-+-+-+-+-+-+-+-	-+-+-+-+-	+-+-+-+-	+-+-+-+-	-+-+-+-+		
Version #	Type	Pa	cket length	1		
+-+-+-+-+-+-+-+-+-	-+-+-+-+-+-	+-+-+-+-+-	+-+-+-+-+-	-+-+-+-+		
Router ID						
+-						
Area ID						
+-						
Checksur	n	1	AuType	1		
+-						
Authentication						
+-						
Authentication						
+-						

OSPF Packet Types

- 1 Hello packet
- 2 Database Description
- 3 Link State Request
- 4 Link State Update
- 5 Link State Acknowledgment

Database Description



Database Description Packet Fields

- Interface MTU
- Options (multicast, external LSAs, etc.)
- Init bit, More bit, Master/Slave bit
- Sequence number: distinguishes DD packets.
- Exchange
 - First packet in an exchange has the I bit sent, all but last have M bit set, sequence number increments on each packet

DD Packet Exchange

- Used to initialize routing state
- Node A sends an empty DD to node B with sequence number n, the I, M, and MS bits set
- Node B responds with a DD with the I and MS bits cleared, seq. no n
 - Can contain LSAs if there are more, set the M bit
- Node A n + 1, I bit cleared, MS bit set, M bit depends on whether there are more LSAs
- Continues until both send cleared M bit

OSPF areas

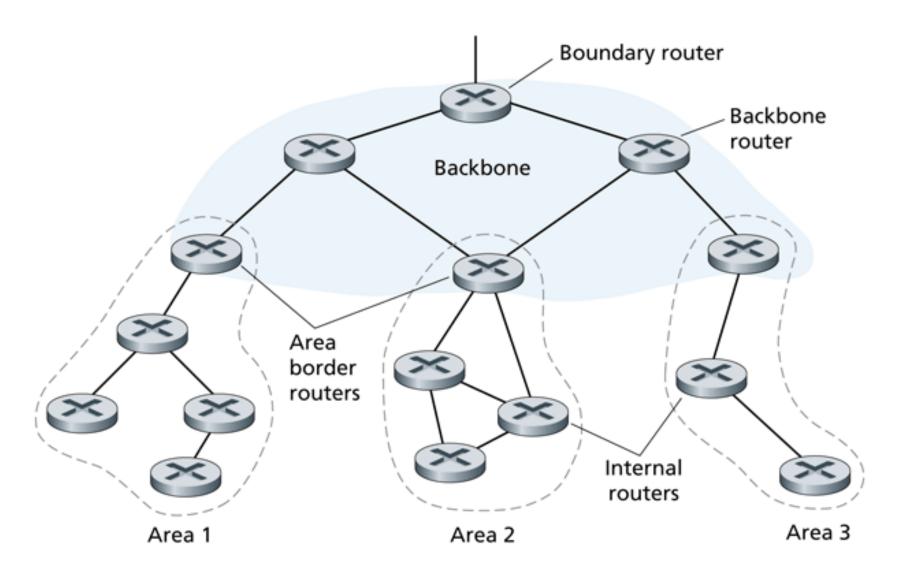


Figure 4.40 • Hierarchically structured OSPF AS with four areas

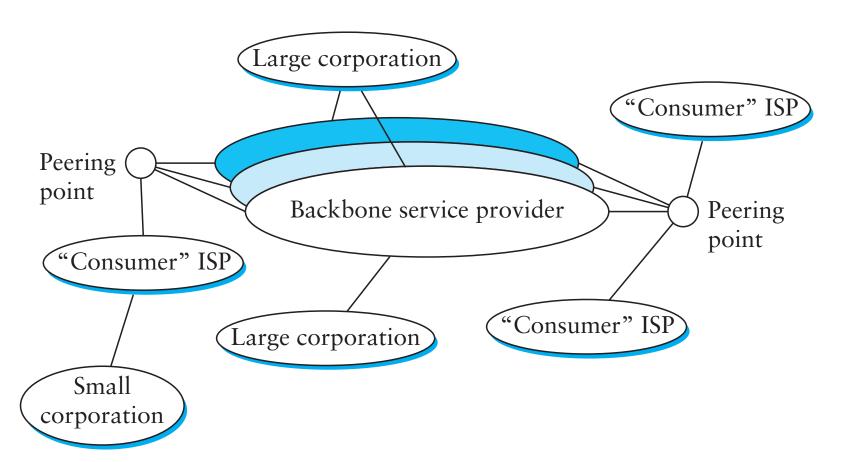
LSA Details

- Many different LSA formats
- Example 1: Router-LSAs, describe a router's links
- Example 2: Summary-LSAs, generated by area border routers
 - Describes an IP network within the area
 - Includes IP address, network mask, and cost metric
- Example 3: AS-external-LSAs
 - Describes an IP network in another AS
 - Includes IP address, netmask, cost, and forwarding address

Basic Algorithms

- Two classes of intra-domain routing algorithms
- Link state (e.g., OSPF)
 - Have a global view of the network
 - Simpler to debug
 - Require global state
- Distance vector (e.g. RIP)
 - Require only local state (less overhead, smaller footprint)
 - Harder to debug
 - Can suffer from loops

The Internet, today



Overview

- Internet structure, ASes
- Forwarding vs. Routing
- Distance vector and link state
- Example distance vector: RIP
- Example link state: OSPF
- Next lecture: BGP