# Review Network Fundamentals

**Jeff Pang**

**15-744 Networking, Spring 2005**

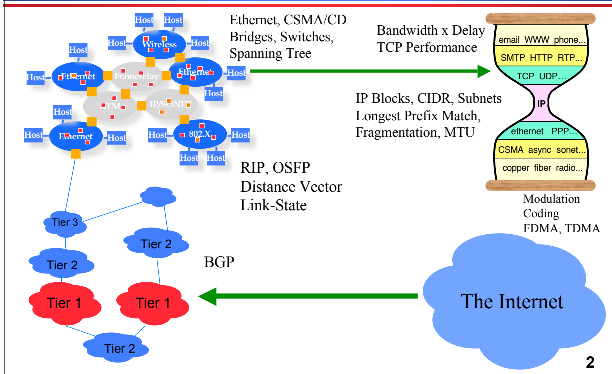
**http://www.cs.cmu.edu/~dga/15-744/S07**

**All slides stolen from Dave and Srini's 15-441 class:**

**http://www.cs.cmu.edu/~srini/15-441/F06/**



## All You Need to Know in 1 Slide






## A More Conventional Overview

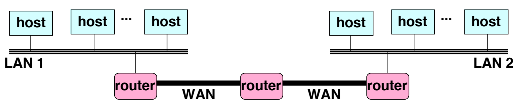
- IP Addressing
- IP Forwarding
- IP Packet Format
- IP Routing
- Performance Calculations
- Link Layer Stuff
- Physical Layer Stuff



# IP Addressing

# What is an Internetwork?

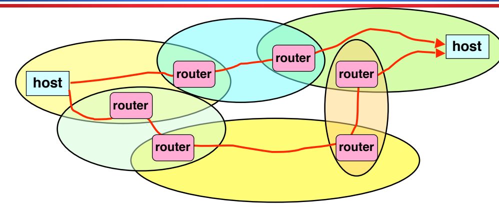
- Multiple incompatible LANs can be physically connected by specialized computers called ***routers***
- The connected networks are called an ***internetwork***
  - The "*Internet*" is one (very big & successful) example of an internetwork




LAN 1 and LAN 2 might be completely different, totally incompatible LANs (e.g., Ethernet and ATM)



# Logical Structure of Internet

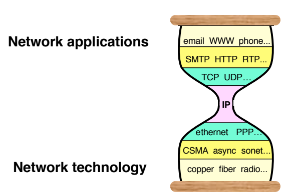



- Ad hoc interconnection of networks
  - No particular topology
  - Vastly different router & link capacities
- Send packets from source to destination by hopping through networks
  - Router connect one network to another
  - Different paths to destination may exist



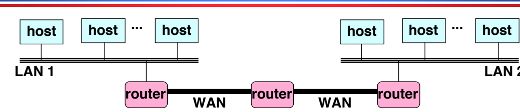
# Internet Protocol (IP)

- Hour Glass Model
  - Create abstraction layer that hides underlying technology from network application software
  - Make as minimal as possible
  - Allows range of current & future technologies
  - Can support many different types of applications





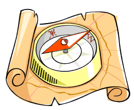

# Problem 3: Internetwork Design




- How do I designate a distant host?
  - Addressing / naming
- How do I send information to a distant host?
  - What gets sent?
  - What route should it take?
- Must support:
  - Heterogeneity LAN technologies
  - Scalability → ensure ability to grow to worldwide scale

## Getting to a Destination

- How do you get driving directions?
- Intersections→routers
- Roads→links/networks
- Roads change slowly




9

## Addressing in IP

- IP addresses are names of interfaces
  - » E.g., 128.2.1.1
- Domain Name System (DNS) names are names of hosts
  - » E.g., www.cmu.edu
- DNS binds host names to interfaces
- Routing binds interface names to paths

10

## Router Table Size

- One entry for every host on the Internet
  - » 440M (7/06) entries,doubling every 2.5 years
- One entry for every LAN
  - » Every host on LAN shares prefix
  - » Still too many and growing quickly
- One entry for every organization
  - » Every host in organization shares prefix
  - » Requires careful address allocation

11

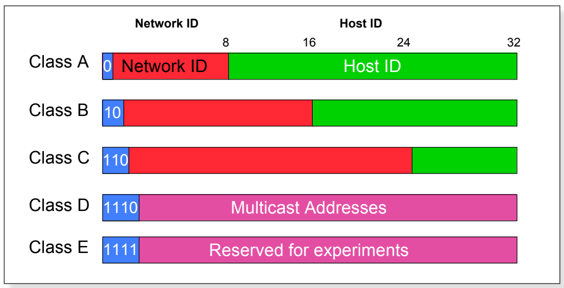
## IP Addresses

- Fixed length: 32 bits
- Initial classful structure (1981) (not relevant now!!!)
- Total IP address size: 4 billion
  - » Class A: 128 networks, 16M hosts
  - » Class B: 16K networks, 64K hosts
  - » Class C: 2M networks, 256 hosts

| High Order Bits | Format | Class |
|---|---|---|
| 0 | 7 bits of net, 24 bits of host | A |
| 10 | 14 bits of net, 16 bits of host | B |
| 110 | 21 bits of net, 8 bits of host | C |

12

## IP Address Classes
### (Some are Obsolete)




13

## Original IP Route Lookup

- Address would specify prefix for forwarding table
  - » Simple lookup
- www.cmu.edu address 128.2.11.43
  - » Class B address – class + network is 128.2
  - » Lookup 128.2 in forwarding table
  - » Prefix – part of address that really matters for routing
- Forwarding table contains
  - » List of class+network entries
  - » A few fixed prefix lengths (8/16/24)
- Large tables
  - » 2 Million class C networks

14

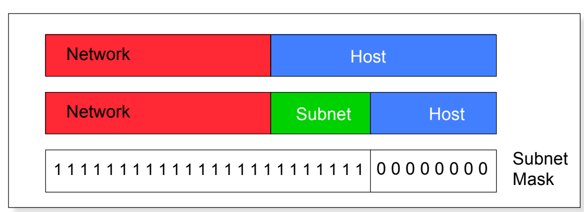
## Subnet Addressing RFC917 (1984)

- Class A & B networks too big
  - » Very few LANs have close to 64K hosts
  - » For electrical/LAN limitations, performance or administrative reasons
- Need simple way to get multiple “networks”
  - » Use bridging, multiple IP networks or split up single network address ranges (subnet)
- CMU case study in RFC
  - » Chose not to adopt – concern that it would not be widely supported ☺

15

## Subnetting

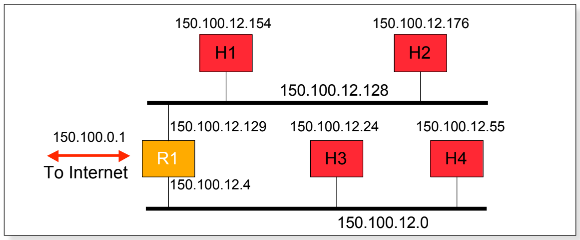
- Add another layer to hierarchy
- Variable length subnet masks
  - » Could subnet a class B into several chunks




16

## Forwarding Example

- Assume a packet arrives with address 150.100.12.176
- Step 1: AND address with class + subnet mask



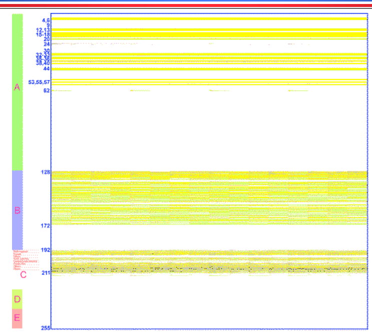

17

## IP Address Problem (1991)

- Address space depletion
  - » In danger of running out of classes A and B
  - » Why?
    - – Class C too small for most domains
    - – Very few class A – very careful about giving them out
    - – Class B – greatest problem
- Class B sparsely populated
  - » But people refuse to give it back
- Large forwarding tables
  - » 2 Million possible class C groups

18

## IP Address Utilization ('97)



http://www.caida.org/outreach/resources/learn/ipv4space/

19

## Important Concepts

- Hierarchical addressing critical for scalable system
  - » Don't require everyone to know everyone else
  - » Reduces number of updates when something changes

20

## Classless Inter-Domain Routing (CIDR) – RFC1338

- Allows arbitrary split between network & host part of address
  - Do not use classes to determine network ID
  - Use common part of address as network number
  - E.g., addresses 192.4.16 - 192.4.31 have the first 20 bits in common. Thus, we use these 20 bits as the network number → 192.4.16/20
- Enables more efficient usage of address space (and router tables) → How?
  - Use single entry for range in forwarding tables
  - Combined forwarding entries when possible

21

## IP Addresses: How to Get One?

- How does an ISP get block of addresses?
  - From Regional Internet Registries (RIRs)
    - ARIN (North America, Southern Africa), APNIC (Asia-Pacific), RIPE (Europe, Northern Africa), LACNIC (South America)
- How about a single host?
  - Hard-coded by system admin in a file
  - DHCP: Dynamic Host Configuration Protocol: dynamically get address: "plug-and-play"
    - Host broadcasts "DHCP discover" msg
    - DHCP server responds with "DHCP offer" msg
    - Host requests IP address: "DHCP request" msg
    - DHCP server sends address: "DHCP ack" msg

22

## IP Forwarding

23

## Host Routing Table Example

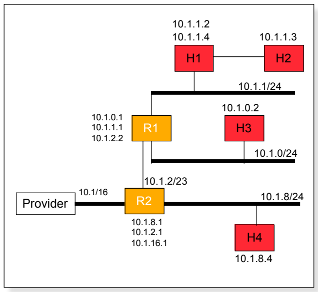
| Destination | Gateway | Genmask | Iface |
|---|---|---|---|
| 128.2.209.100 | 0.0.0.0 | 255.255.255.255 | eth0 |
| 128.2.0.0 | 0.0.0.0 | 255.255.0.0 | eth0 |
| 127.0.0.0 | 0.0.0.0 | 255.0.0.0 | lo |
| 0.0.0.0 | 128.2.254.36 | 0.0.0.0 | eth0 |

- From "netstat –rn"
- Host 128.2.209.100 when plugged into CS ethernet
- Dest 128.2.209.100 → routing to same machine
- Dest 128.2.0.0 → other hosts on same ethernet
- Dest 127.0.0.0 → special loopback address
- Dest 0.0.0.0 → default route to rest of Internet
  - Main CS router: gigrouter.net.cs.cmu.edu (128.2.254.36)

24

## Routing to the Network

- Packet to 10.1.1.3 arrives
- Path is R2 – R1 – H1 – H2



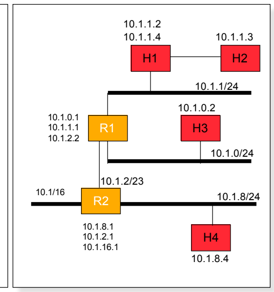



## Routing Within the Subnet

- Packet to 10.1.1.3
- Matches 10.1.0.0/23

Routing table at R2

| Destination | Next Hop | Interface |
|---|---|---|
| 127.0.0.1 | 127.0.0.1 | lo0 |
| Default or 0/0 | provider | 10.1.16.1 |
| 10.1.8.0/24 | 10.1.8.1 | 10.1.8.1 |
| 10.1.2.0/23 | 10.1.2.1 | 10.1.2.1 |
| 10.1.0.0/23 | 10.1.2.2 | 10.1.2.1 |



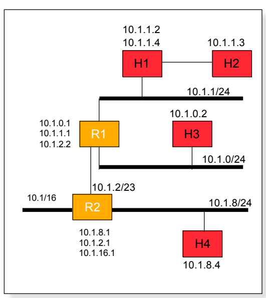



## Routing Within the Subnet

- Packet to 10.1.1.3
- Matches 10.1.1.1/31
  - Longest prefix match

Routing table at R1

| Destination | Next Hop | Interface |
|---|---|---|
| 127.0.0.1 | 127.0.0.1 | lo0 |
| Default or 0/0 | 10.1.2.1 | 10.1.2.2 |
| 10.1.0.0/24 | 10.1.0.1 | 10.1.0.1 |
| 10.1.1.0/24 | 10.1.1.1 | 10.1.1.4 |
| 10.1.2.0/23 | 10.1.2.2 | 10.1.2.2 |
| 10.1.1.2/31 | 10.1.1.2 | 10.1.1.2 |



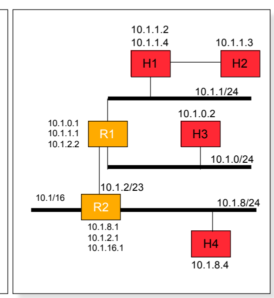



## Routing Within the Subnet

- Packet to 10.1.1.3
- Direct route
  - Longest prefix match

Routing table at H1

| Destination | Next Hop | Interface |
|---|---|---|
| 127.0.0.1 | 127.0.0.1 | lo0 |
| Default or 0/0 | 10.1.1.1 | 10.1.1.2 |
| 10.1.1.0/24 | 10.1.1.2 | 10.1.1.1 |
| 10.1.1.3/31 | 10.1.1.2 | 10.1.1.2 |



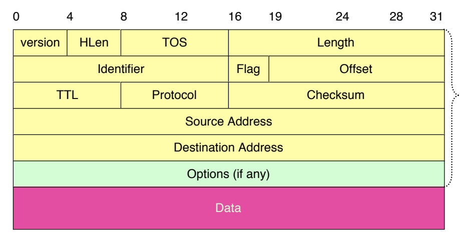
# IP Packet Format

29

# IP Service Model

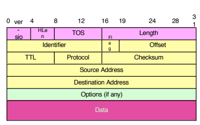
- Low-level communication model provided by Internet
- Datagram
  - » Each packet self-contained
    - – All information needed to get to destination
    - – No advance setup or connection maintenance
  - » Analogous to letter or telegram

IPv4 Packet Format




30

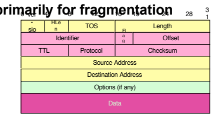
# IPv4 Header Fields



- Version: IP Version
  - » 4 for IPv4
- HLen: Header Length
  - » 32-bit words (typically 5)
- TOS: Type of Service
  - » Priority information
- Length: Packet Length
  - » Bytes (including header)
- Header format can change with versions
  - » First byte identifies version
- Length field limits packets to 65,535 bytes
  - » In practice, break into much smaller packets for network performance considerations

31

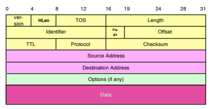
# IPv4 Header Fields



- Identifier, flags, fragment offset → used primarily for fragmentation
- Time to live
  - » Must be decremented at each router
  - » Packets with TTL=0 are thrown away
  - » Ensure packets exit the network
- Protocol
  - » Demultiplexing to higher layer protocols
  - » TCP = 6, ICMP = 1, UDP = 17…
- Header checksum
  - » Ensures some degree of header integrity
  - » Relatively weak – 16 bit
- Options
  - » E.g. Source routing, record route, etc.
  - » Performance issues
    - – Poorly supported

32

## IPv4 Header Fields

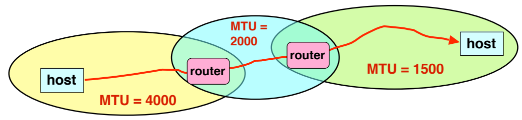



- Source Address
  - » 32-bit IP address of sender
- Destination Address
  - » 32-bit IP address of destination

- Like the addresses on an envelope
- Globally unique identification of sender & receiver



## IP Fragmentation




- Every network has own Maximum Transmission Unit (MTU)
  - » Largest IP datagram it can carry within its own packet frame
    - – E.g., Ethernet is 1500 bytes
  - » Don't know MTUs of all intermediate networks in advance
- IP Solution
  - » When hit network with small MTU, fragment packets



## Reassembly

- Where to do reassembly?
  - » End nodes or at routers?

- End nodes
  - » Avoids unnecessary work where large packets are fragmented multiple times
  - » If any fragment missing, delete entire packet

- Dangerous to do at intermediate nodes
  - » How much buffer space required at routers?
  - » What if routes in network change?
    - – Multiple paths through network
    - – All fragments only required to go through destination



## Fragmentation and Reassembly Concepts

- Demonstrates many Internet concepts
- Decentralized
  - » Every network can choose MTU
- Connectionless
  - » Each (fragment of) packet contains full routing information
  - » Fragments can proceed independently and along different routes
- Best effort
  - » Fail by dropping packet
  - » Destination can give up on reassembly
  - » No need to signal sender that failure occurred
- Complex endpoints and simple routers
  - » Reassembly at endpoints

## Fragmentation is Harmful

- Uses resources poorly
  - Forwarding costs per packet
  - Best if we can send large chunks of data
  - Worst case: packet just bigger than MTU
- Poor end-to-end performance
  - Loss of a fragment
- Path MTU discovery protocol → determines minimum MTU along route
  - Uses ICMP error messages
- Common theme in system design
  - Assure correctness by implementing complete protocol
  - Optimize common cases to avoid full complexity

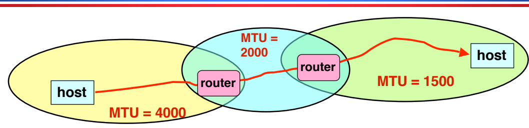


## Internet Control Message Protocol (ICMP)

- Short messages used to send error & other control information
- Examples
  - Ping request / response
    - Can use to check whether remote host reachable
  - Destination unreachable
    - Indicates how packet got & why couldn't go further
  - Flow control
    - Slow down packet delivery rate
  - Redirect
    - Suggest alternate routing path for future messages
  - Router solicitation / advertisement
    - Helps newly connected host discover local router
  - Timeout
    - Packet exceeded maximum hop limit



## IP MTU Discovery with ICMP




- Typically send series of packets from one host to another
- Typically, all will follow same route
  - Routes remain stable for minutes at a time
- Makes sense to determine path MTU before sending real packets
- Operation
  - Send max-sized packet with "do not fragment" flag set
  - If encounters problem, ICMP message will be returned
    - "Destination unreachable: Fragmentation needed"
    - Usually indicates MTU encountered



## Important Concepts

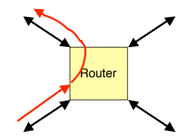
- Base-level protocol (IP) provides minimal service level
  - Allows highly decentralized implementation
  - Each step involves determining next hop
  - Most of the work at the endpoints
- ICMP provides low-level error reporting
- IP forwarding → global addressing, alternatives, lookup tables
- IP addressing → hierarchical, CIDR
- IP service → best effort, simplicity of routers
- IP packets → header fields, fragmentation, ICMP

# IP Routing

41

# IP Forwarding

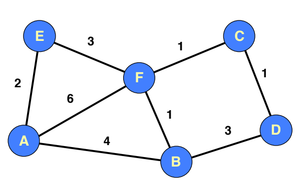
- The Story So Far…
  - » IP addresses are structure to reflect Internet structure
  - » IP packet headers carry these addresses
  - » When Packet Arrives at Router
    - – Examine header to determine intended destination
    - – *Look up in table to determine next hop in path*
    - – Send packet out appropriate port
- This/next lecture
  - » How to generate the forwarding table




42

# Graph Model

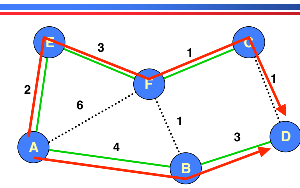
- Represent each router as node
- Direct link between routers represented by edge
  - » Symmetric links ⇒ undirected graph
- Edge "cost" c(x,y) denotes measure of difficulty of using link
  - » delay, $ cost, or congestion level
- Task
  - » Determine least cost path from every node to every other node
    - – Path cost d(x,y) = sum of link costs




43

# Routes from Node A

| Forwarding Table for A | | |
|---|---|---|
| Dest | Cost | Next Hop |
| A | 0 | A |
| B | 4 | B |
| C | 6 | E |
| D | 7 | B |
| E | 2 | E |
| F | 5 | E |




- Properties
  - » Some set of shortest paths forms tree
    - – Shortest path spanning tree
  - » Solution not unique
    - – E.g., A-E-F-C-D also has cost 7

44

## Ways to Compute Shortest Paths

- Centralized
  - » Collect graph structure in one place
  - » Use standard graph algorithm
  - » Disseminate routing tables
- Link-state
  - » Every node collects complete graph structure
  - » Each computes shortest paths from it
  - » Each generates own routing table
- Distance-vector
  - » No one has copy of graph
  - » Nodes construct their own tables iteratively
  - » Each sends information about its table to neighbors

45

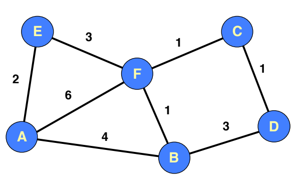
## Outline

- Distance Vector
- Link State
- Routing Hierarchy
- BGP

46

## Distance-Vector Method

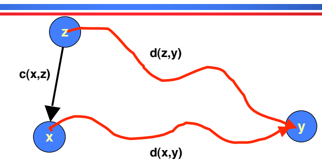
| Initial Table for A | | |
|---|---|---|
| Dest | Cost | Next Hop |
| A | 0 | A |
| B | 4 | B |
| C | ∞ | – |
| D | ∞ | – |
| E | 2 | E |
| F | 6 | F |




- Idea
  - » At any time, have cost/next hop of best known path to destination
  - » Use cost ∞ when no path known
- Initially
  - » Only have entries for directly connected nodes

47

## Distance-Vector Update




- Update(x,y,z)

```
d ← c(x,z) + d(z,y)          # Cost of path from x to y with first hop z
if d < d(x,y)
    # Found better path
    return d,z               # Updated cost / next hop
else
    return d(x,y), nexthop(x,y)        # Existing cost / next hop
```

48

# Algorithm

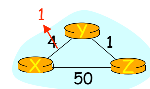
- **Bellman-Ford algorithm**
- **Repeat**
  - **For every node x**
    - **For every neighbor z**
      - **For every destination y**
        - **d(x,y) ← Update(x,y,z)**
- **Until converge**



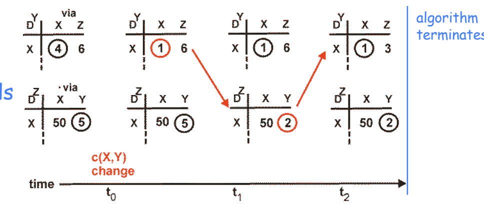
# Distance Vector: Link Cost Changes

**Link cost changes:**

- **Node detects local link cost change**
- **Updates distance table**
- **If cost change in least cost path, notify neighbors**




"good news travels fast"

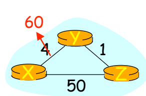


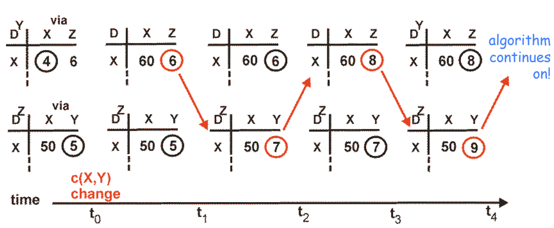



# Distance Vector: Link Cost Changes

**Link cost changes:**

- **Good news travels fast**
- **Bad news travels slow - "count to infinity" problem!**



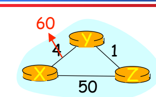



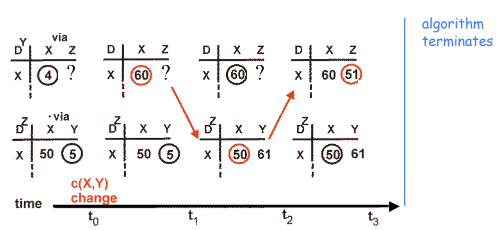



# Distance Vector: Split Horizon

**If Z routes through Y to get to X :**

- **Z does not advertise its route to X back to Y**

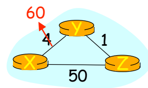


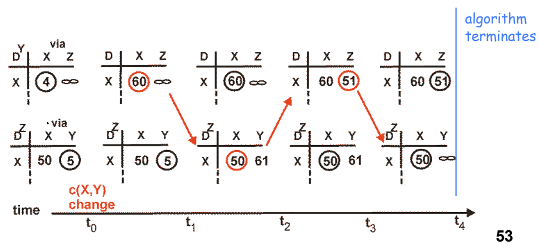



## Distance Vector: Poison Reverse

If Z routes through Y to get to X :

- Z tells Y its (Z's) distance to X is infinite (so Y won't route to X via Z)
- Eliminates some possible timeouts with split horizon
- Will this completely solve count to infinity problem?




53

## Poison Reverse Failures

| Table for A | | | Table for B | | | Table for D | | | Table for F | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Dst | Cst | Hop | Dst | Cst | Hop | Dst | Cst | Hop | Dst | Cst | Hop |
| C | 7 | F | C | 8 | A | C | 9 | B | C | 1 | C |

Table for A — Forced Update

| Dst | Cst | Hop |
|---|---|---|
| C | ∞ | – |

Table for F — Forced Update

| Dst | Cst | Hop |
|---|---|---|
| C | ∞ | – |

Table for A — Better Route

| Dst | Cst | Hop |
|---|---|---|
| C | 13 | D |

Table for B — Forced Update

| Dst | Cst | Hop |
|---|---|---|
| C | 14 | A |

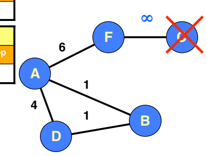
Table for D — Forced Update

| Dst | Cst | Hop |
|---|---|---|
| C | 15 | B |

Table for A — Forced Update

| Dst | Cst | Hop |
|---|---|---|
| C | 19 | D |

. . .




- Iterations don't converge
- "Count to infinity"
- Solution
  - » Make "infinity" smaller
  - » What is upper bound on maximum path length?

54

## Routing Information Protocol (RIP)

- Earliest IP routing protocol (1982 BSD)
  - » Current standard is version 2 (RFC 1723)
- Features
  - » Every link has cost 1
  - » "Infinity" = 16
    - – Limits to networks where everything reachable within 15 hops
- Sending Updates
  - » Every router listens for updates on UDP port 520
  - » RIP message can contain entries for up to 25 table entries

55

## Outline

- Distance Vector
- Link State
- Routing Hierarchy
- BGP

56

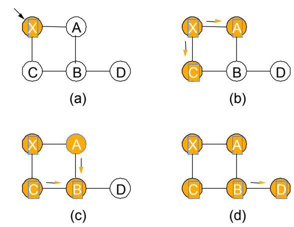
## Link State Protocol Concept

- **Every node gets complete copy of graph**
  - » **Every node "floods" network with data about its outgoing links**
- **Every node computes routes to every other node**
  - » **Using single-source, shortest-path algorithm**
- **Process performed whenever needed**
  - » **When connections die / reappear**

57

## Sending Link States by Flooding

- **X Wants to Send Information**
  - » **Sends on all outgoing links**
- **When Node Y Receives Information from Z**
  - » **Send on all links other than Z**

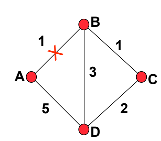



58

## Link State Characteristics

- **With consistent LSDBs*, all nodes compute consistent loop-free paths**
- **Can still have transient loops**

*Link State Data Base




Packet from C→A
may loop around BDC
if B knows about failure
and C & D do not

59

## OSPF Routing Protocol

- **Open**
  - » **Open standard created by IETF**
- **Shortest-path first**
  - » **Another name for Dijkstra's algorithm**
- **More prevalent than RIP**

60

## Flooding Issues

- When should it be performed
  - Periodically
  - When status of link changes
    - Detected by connected node
- What happens when router goes down & back up
  - Sequence number reset to 0
    - Other routers may have entries with higher sequence numbers
  - Router will send out LSAs with number 0
  - Will get back LSAs with last valid sequence number p
  - Router sets sequence number to p+1 & resends

61

## Adoption of OSPF

- RIP viewed as outmoded
  - Good when networks small and routers had limited memory & computational power
- OSPF Advantages
  - Fast convergence when configuration changes

62

## Comparison of LS and DV Algorithms

**Message complexity**

- LS: with n nodes, E links, O(nE) messages
- DV: exchange between neighbors only

**Speed of Convergence**

- LS: Complex computation
  - But…can forward before computation
  - may have oscillations
- DV: convergence time varies
  - may be routing loops
  - count-to-infinity problem
  - (faster with triggered updates)

**Space requirements:**

- LS maintains entire topology
- DV maintains only neighbor state

63

## Comparison of LS and DV Algorithms

Robustness: what happens if router malfunctions?

LS:

- node can advertise incorrect *link* cost
- each node computes only its *own* table

DV:

- DV node can advertise incorrect *path* cost
- each node's table used by others
  - errors propagate thru network
- Other tradeoffs
  - Making LSP flood reliable

64

## Outline

- Distance Vector
- Link State
- Routing Hierarchy
- BGP

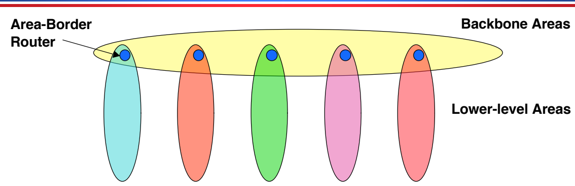
65

## Routing Hierarchies

- Flat routing doesn't scale
  - » Storage → Each node cannot be expected to store routes to every destination (or destination network)
  - » Convergence times increase
  - » Communication → Total message count increases
- Key observation
  - » Need less information with increasing distance to destination
  - » Need lower diameters networks
- Solution: area hierarchy

66

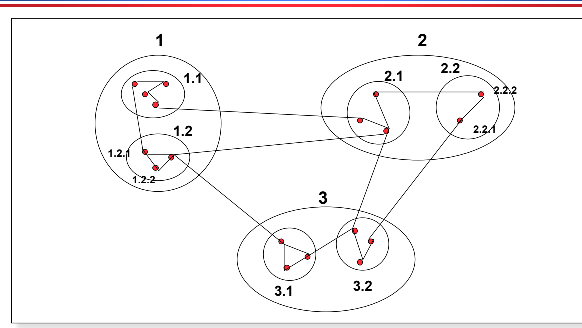
## Routing Hierarchy




- Partition Network into "Areas"
  - » Within area
    - – Each node has routes to every other node
  - » Outside area
    - – Each node has routes for other top-level areas only
    - – Inter-area packets are routed to nearest appropriate border router
- Constraint: no path between two sub-areas of an area can exit that area

67

## Area Hierarchy Addressing




68

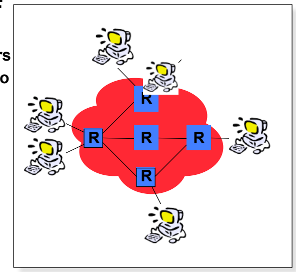
## Outline

- Distance Vector
- Link State
- Routing Hierarchy
- BGP

69

## A Logical View of the Internet?

- After looking at RIP/OSPF descriptions
  - » End-hosts connected to routers
  - » Routers exchange messages to determine connectivity
- NOT TRUE!




70

## Internet's Area Hierarchy

- What is an Autonomous System (AS)?
  - » A set of routers under a single technical administration, using an *interior gateway protocol (IGP)* and common metrics to route packets within the AS and using an *exterior gateway protocol (EGP)* to route packets to other AS's
- Each AS assigned unique ID
- AS's peer at network exchanges

71

## AS Numbers (ASNs)

ASNs are 16 bit values 64512 through 65535 are "private"

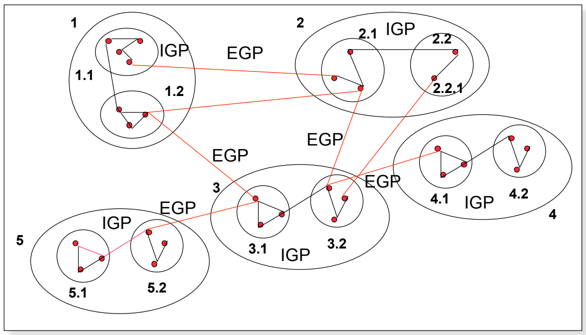
Currently over 15,000 in use

- Genuity: 1
- MIT: 3
- CMU: 9
- UC San Diego: 7377
- AT&T: 7018, 6341, 5074, …
- UUNET: 701, 702, 284, 12199, …
- Sprint: 1239, 1240, 6211, 6242, …
- …

ASNs represent units of routing policy

72

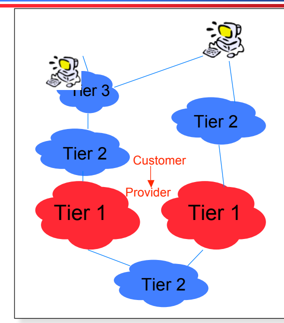
## Example




73

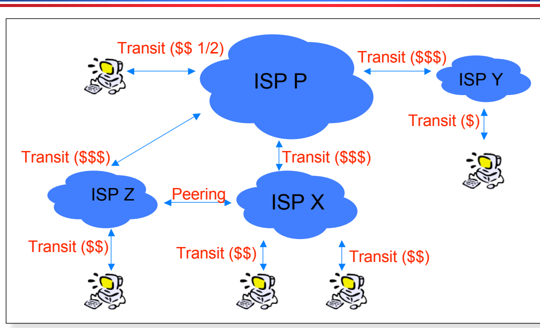
## A Logical View of the Internet

- Tier 1 ISP
  - » "Default-free" with global reachability info
- Tier 2 ISP
  - » Regional or country-wide
- Tier 3 ISP
  - » Local




74

## Transit vs. Peering




75

## Policy Impact

- "Valley-free" routing
  - » Number links as (+1, 0, -1) for provider, peer and customer
  - » In any path should only see sequence of +1, followed by at most one 0, followed by sequence of -1
- WHY?
  - » Consider the economics of the situation

76

# External BGP (E-BGP)

77

# Choices

- Link state or distance vector?
  - No universal metric – policy decisions
- Problems with distance-vector:
  - Bellman-Ford algorithm may not converge
- Problems with link state:
  - Metric used by routers not the same – loops
  - LS database too large – entire Internet
  - May expose policies to other AS's

78

# Solution: Distance Vector with Path

- Each routing update carries the entire path
- Loops are detected as follows:
  - When AS gets route, check if AS already in path
    - If yes, reject route
    - If no, add self and (possibly) advertise route further
- Advantage:
  - Metrics are local - AS chooses path, protocol ensures no loops

79

# Interconnecting BGP Peers

- BGP uses TCP to connect peers
- Advantages:
  - Simplifies BGP
  - No need for periodic refresh - routes are valid until withdrawn, or the connection is lost
  - Incremental updates
- Disadvantages
  - Congestion control on a routing protocol?
  - Poor interaction during high load

80

## Hop-by-hop Model

- BGP advertises to neighbors only those routes that it uses
  - » Consistent with the hop-by-hop Internet paradigm
  - » e.g., AS1 cannot tell AS2 to route to other AS's in a manner different than what AS2 has chosen (need source routing for that)
- BGP enforces policies by choosing paths from multiple alternatives and controlling advertisement to other AS's

81

## Examples of BGP Policies

- A multi-homed AS refuses to act as transit
  - » Limit path advertisement
- A multi-homed AS can become transit for some AS's
  - » Only advertise paths to some AS's
- An AS can favor or disfavor certain AS's for traffic transit from itself

82

## BGP UPDATE Message

- List of withdrawn routes
- Network layer reachability information
  - » List of reachable prefixes
- Path attributes
  - » Origin
  - » Path
  - » Metrics
- All prefixes advertised in message have same path attributes

83

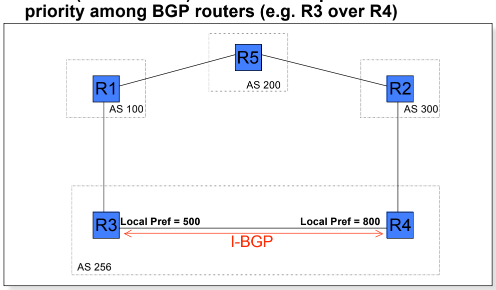
## Path Selection Criteria

- Attributes + external (policy) information
- Examples:
  - » Hop count
  - » Policy considerations
    - – Preference for AS
    - – Presence or absence of certain AS
  - » Path origin
  - » Link dynamics

84

# LOCAL PREF

- Local (within an AS) mechanism to provide relative priority among BGP routers (e.g. R3 over R4)




85

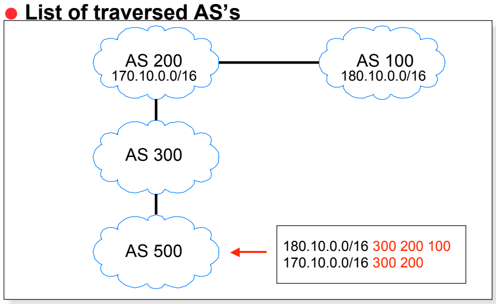
# LOCAL PREF – Common Uses

- Peering vs. transit
  - » Prefer to use peering connection, why?
- In general, customer > peer > provider
  - » Use LOCAL PREF to ensure this

86

# AS_PATH

- List of traversed AS's




87

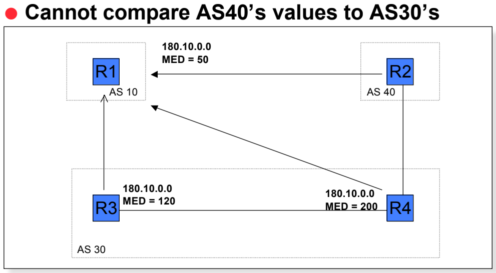
# Multi-Exit Discriminator (MED)

- Hint to external neighbors about the preferred path into an AS
  - » Non-transitive attribute
    - – Different AS choose different scales
- Used when two AS's connect to each other in more than one place

88

## MED

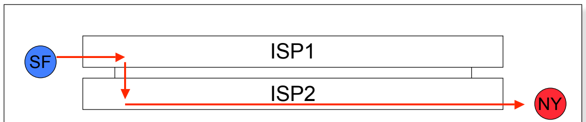
- Hint to R1 to use R3 over R4 link
- Cannot compare AS40's values to AS30's




89

## MED

- MED is typically used in provider/subscriber scenarios
- It can lead to unfairness if used between ISP because it may force one ISP to carry more traffic:




- ISP1 ignores MED from ISP2
- ISP2 obeys MED from ISP1
- ISP2 ends up carrying traffic most of the way

90

## Decision Process

- Processing order of attributes:
  - » Select route with highest LOCAL-PREF
  - » Select route with shortest AS-PATH
  - » Apply MED (if routes learned from same neighbor)

91

## Important Concepts

- Wide area Internet structure and routing driven by economic considerations
  - » Customer, providers and peers
- BGP designed to:
  - » Provide hierarchy that allows scalability
  - » Allow enforcement of policies related to structure
- Mechanisms
  - » Path vector – scalable, hides structure from neighbors, detects loops quickly

92

# Performance



# Outline

- Flow and Error Control
- TCP Performance



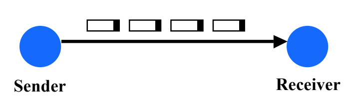
# Flow Control and Error Control

- Naïve protocol.
- Dealing with receiver overflow: flow control.
- Dealing with packet loss and corruption: error control.
- Meta-comment: these issues are relevant at many layers.
  - » Link layer: sender and receiver attached to the same "wire"
  - » End-to-end: transmission control protocol (TCP) - sender and receiver are the end points of a connection
- How can we implement flow control?
  - » "You may send" (windows, stop-and-wait, etc.)
  - » "Please shut up" (source quench, 802.3x pause frames, etc.)
  - » Where are each of these appropriate?



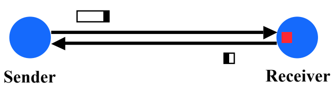
# A Naïve Protocol

- Sender simply sends to the receiver whenever it has packets.
- Potential problem: sender can outrun the receiver.
  - » Receiver too slow, buffer overflow, ..
- Not always a problem: receiver might be fast enough.



## Adding Flow Control

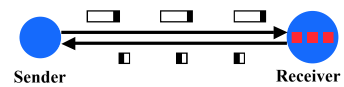
- Stop and wait flow control: sender waits to send the next packet until the previous packet has been acknowledged by the receiver.
  - » Receiver can pace the receiver
- Drawbacks: adds overheads, slowdown for long links.




97

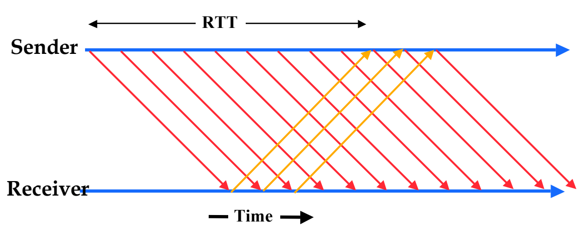
## Window Flow Control

- Stop and wait flow control results in poor throughput for long-delay paths: packet size/ roundtrip-time.
- Solution: receiver provides sender with a window that it can fill with packets.
  - » The window is backed up by buffer space on receiver
  - » Receiver acknowledges the a packet every time a packet is consumed and a buffer is freed




98

## Bandwidth-Delay Product

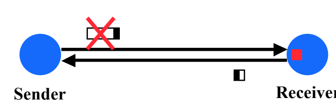



$$\text{Max Throughput} = \frac{\text{Window Size}}{\text{Roundtrip Time}}$$

99

## Dealing with Errors Stop and Wait Case

- Packets can get lost, corrupted, or duplicated.
  - » Error detection or correction turns corrupted packet in lost or correct packet
- Duplicate packet: use sequence numbers.
- Lost packet: time outs and acknowledgements.
  - » Positive versus negative acknowledgements
  - » Sender side versus receiver side timeouts
- Window based flow control: more aggressive use of sequence numbers (see transport lectures).




100

## What is Used in Practice?

- No flow or error control.
  - E.g. regular Ethernet, just uses CRC for error detection
- Flow control only.
  - E.g. Gigabit Ethernet
- Flow and error control.
  - E.g. X.25 (older connection-based service at 64 Kbs that guarantees reliable in order delivery of data)

101

## Outline

- Flow and Error Control
- TCP Performance

102

## TCP Performance

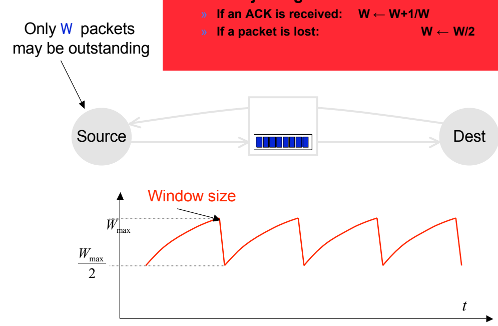
- Can TCP saturate a link?
- Congestion control
  - Increase utilization until… link becomes congested
  - React by decreasing window by 50%
  - Window is proportional to rate * RTT
- Doesn't this mean that the network oscillates between 50 and 100% utilization?
  - Average utilization = 75%??
  - No…this is *not* right!

103

## TCP Congestion Control

**Rule for adjusting W**

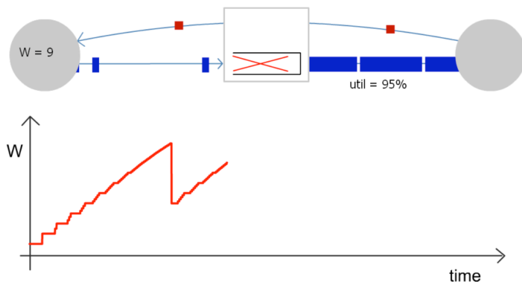
- If an ACK is received: W ← W+1/W
- If a packet is lost: W ← W/2




104

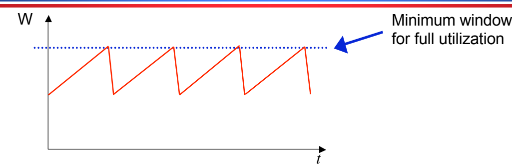
# Single TCP Flow

Router without buffers




105

# Summary Unbuffered Link




- The router can't fully utilize the link
  - If the window is too small, link is not full
  - If the link is full, next window increase causes drop
  - With no buffer it still achieves 75% utilization

106

# TCP Performance

- In the real world, router queues play important role
  - Window is proportional to rate * RTT
    - But, RTT changes as well the window
  - Window to fill links = propagation RTT * bottleneck bandwidth
    - If window is larger, packets sit in queue on bottleneck link

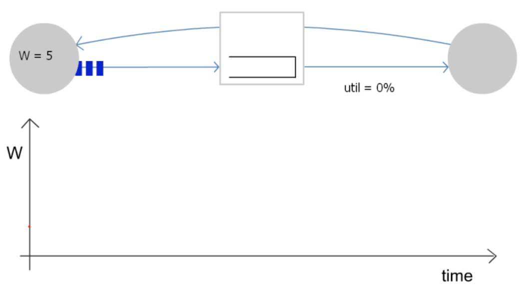
107

# TCP Performance

- If we have a large router queue → can get 100% utilization
  - But, router queues can cause large delays
- How big does the queue need to be?
  - Windows vary from W → W/2
    - Must make sure that link is always full
    - W/2 > RTT * BW
    - W = RTT * BW + Qsize
    - Therefore, Qsize > RTT * BW
  - Ensures 100% utilization
  - Delay?
    - Varies between RTT and 2 * RTT

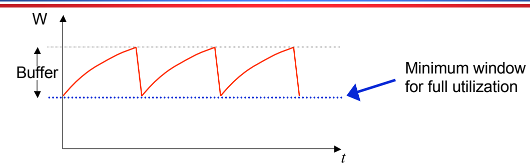
108

## Single TCP Flow
### Router with large enough buffers for full link utilization






## Summary Buffered Link




- With sufficient buffering we achieve full link utilization
  - » The window is always above the critical threshold
  - » Buffer absorbs changes in window size
    - – Buffer Size = Height of TCP Sawtooth
    - – Minimum buffer size needed is 2T*C
  - » This is the origin of the rule-of-thumb



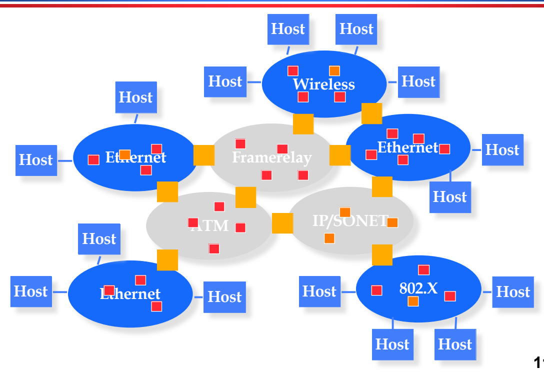
## Link-Layer



## Outline

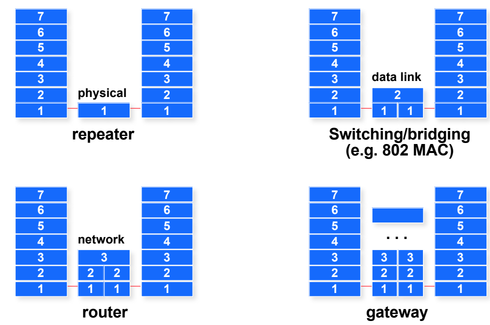
- Switching
- Ethernet and CSMA/CD
- Bridging and Spanning Tree

## An Inter-network




113

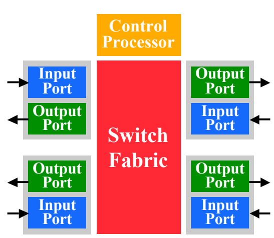
## Internetworking Options




114

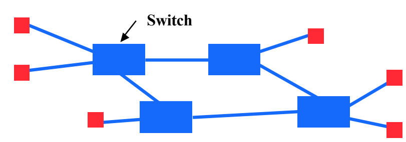
## Switch Architecture

- Takes in packets in one interface and has to forward them to an output interface based on the address.
  - » A big intersection
  - » Same idea for bridges, switches, routers: address look up differs
- Control processor manages the switch and executes higher level protocols.
  - » E.g. routing, management, ..
- The switch fabric directs the traffic to the right output port.
- The input and output ports deal with transmission and reception of packets.




115

## Packet Forwarding: Address Lookup




| Address | Next Hop | Info |
|---|---|---|
| B31123812508 | 3 | 13 |
| 38913C3C2137 | 3 | - |
| A21023C90590 | 0 | - |
| | | |
| 128.2.15.3 | 1 | (2,34) |

- Address from header.
  - » Absolute address (e.g. Ethernet)
  - » (IP address for routers)
  - » (VC identifier, e.g. ATM))
- Next hop: output port for packet.
- Info: priority, VC id, ..
- Table is filled in by routing protocol.

116

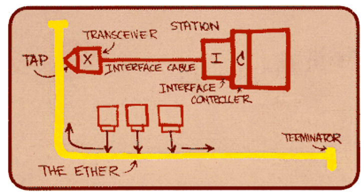
## Outline

- Switching
- Ethernet and CSMA/CD
- Bridging and Spanning Tree



## Ethernet

- First practical local area network, built at Xerox PARC in 70's
- "Dominant" LAN technology:
  - » Cheap
  - » Kept up with speed race: 10, 100, 1000 Mbps

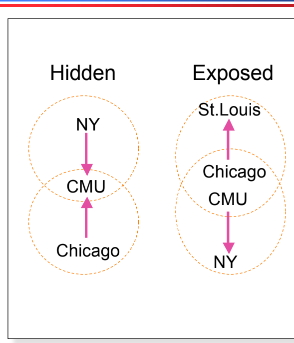



Metcalfe's Ethernet sketch



## Ethernet MAC – Carrier Sense

- Basic idea:
  - » Listen to wire before transmission
  - » Avoid collision with active transmission
- Why doesn't wireless have this?
  - » In wireless, relevant contention at the receiver, not sender
    - – Hidden terminal
    - – Exposed terminal



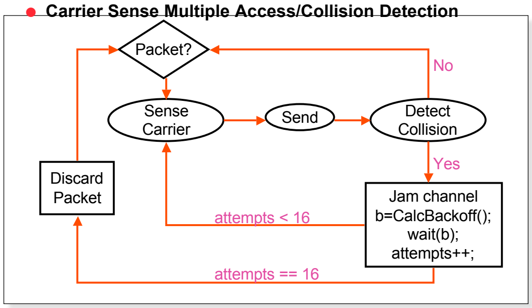



## Ethernet MAC – Collision Detection

- Basic idea:
  - » Listen while transmitting
  - » If you notice interference → assume collision
- Why doesn't wireless have this?
  - » Very difficult for radios to listen and transmit
  - » Signal strength is reduced by distance for radio
    - – Much easier to hear "local, powerful" radio station than one in NY
    - – You may not notice any "interference"

# Ethernet MAC (CSMA/CD)

- Carrier Sense Multiple Access/Collision Detection




121

# Ethernet's CSMA/CD (more)

**Jam Signal:** make sure all other transmitters are aware of collision; 48 bits;

**Exponential Backoff:**

- If deterministic delay after collision, collision will occur again in lockstep
- Why not random delay with fixed mean?
  - Few senders → needless waiting
  - Too many senders → too many collisions
- *Goal*: adapt retransmission attempts to estimated current load
  - heavy load: random wait will be longer

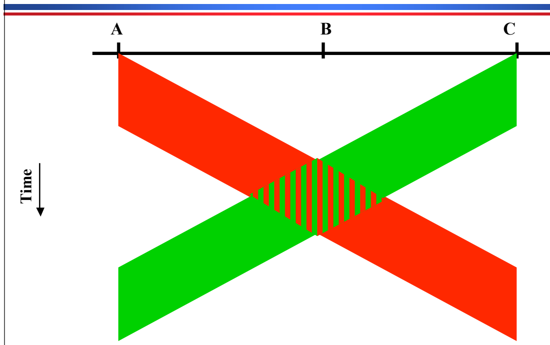
122

# Ethernet Backoff Calculation

- Exponentially increasing random delay
  - Infer senders from # of collisions
  - More senders → increase wait time
- First collision: choose K from {0,1}; delay is K x 512 bit transmission times
- After second collision: choose K from {0,1,2,3}…
- After ten or more collisions, choose K from {0,1,2,3,4,…,1023}

123

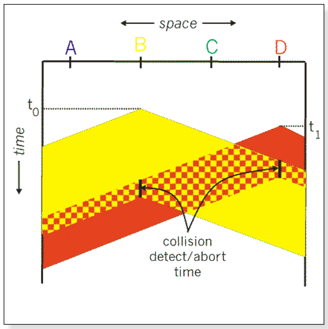
# Collisions




124

## Minimum Packet Size

- What if two people sent really small packets
  - » How do you find collision?




125

## Ethernet Collision Detect

- Min packet length > 2x max prop delay
  - » If A, B are at opposite sides of link, and B starts one link prop delay after A
- Jam network for 32-48 bits after collision, then stop sending
  - » Ensures that everyone notices collision

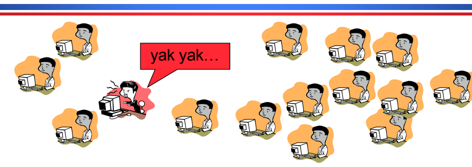
126

## Outline

- Switching
- Ethernet and CDMA
- Bridging and Spanning Tree

127

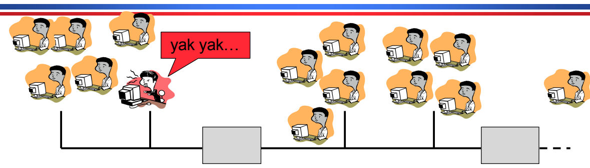
## Scale




- What breaks when we keep adding people to the same wire?

128

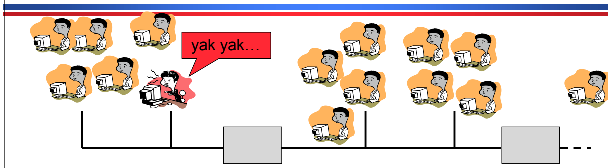
## Scale




- What breaks when we keep adding people to the same wire?
- Only solution: split up the people onto multiple wires
  - » But how can they talk to each other?



## Problem 1 – Reconnecting LANs

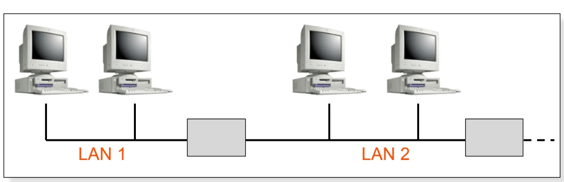



- When should these boxes forward packets between wires?
- How do you specify a destination?
- How does your packet find its way?



## Building Larger LANs: Bridges

- Extend reach of a single shared medium
- Connect two or more "segments" by copying data frames between them
  - » Only copy data when needed → key difference from repeaters/hubs
  - » Reduce collision domain compared with single LAN
  - » Separate segments can send at once → much greater bandwidth
- Challenge: learning which packets to copy across links

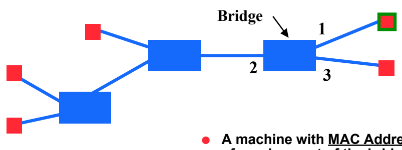





## Transparent Bridges

- Design goals:
  - » Self-configuring without hardware or software changes
  - » Bridge do not impact the operation of the individual LANs
- Three parts to making bridges transparent:
  - Forwarding frames
  - Learning addresses/host locations
  - Spanning tree algorithm

## Frame Forwarding



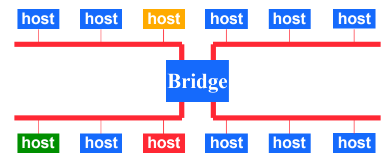

| MAC Address | Port | Age |
|---|---|---|
| A21032C9A591 | 1 | 36 |
| 99A323C90842 | 2 | 01 |
| 8711C98900AA | 2 | 15 |
| 301B2369011C | 2 | 16 |
| 695519001190 | 3 | 11 |

- A machine with MAC Address lies in the direction of number port of the bridge
- For every packet, the bridge "looks up" the entry for the packets destination MAC address and forwards the packet on that port.
  - Other packets are broadcast – why?
- Timer is used to flush old entries

133

## Learning Bridges

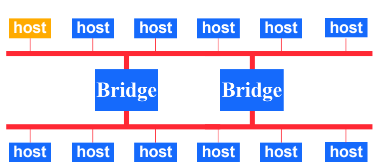
- Manually filling in bridge tables?
  - Time consuming, error-prone
- Keep track of source address of packets arriving on every link, showing what segment hosts are on
  - Fill in the forwarding table based on this information




134

## Spanning Tree Bridges

- More complex topologies can provide redundancy.
  - But can also create loops.
- What is the problem with loops?
- Solution: spanning tree




135

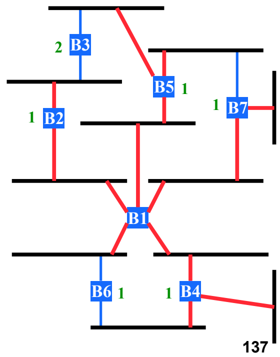
## Spanning Tree Protocol Overview

Embed a tree that provides a single unique path to each destination:
- Elect a single bridge as a root bridge
- Each bridge calculates the distance of the shortest path to the root bridge
- Each LAN identifies a *designated bridge*, the bridge closest to the root. It will forward packets to the root.
- Each bridge determines a *root port*, which will be used to send packets to the root
- Identify the ports that form the spanning tree

136

# Spanning Tree Algorithm
# Steps

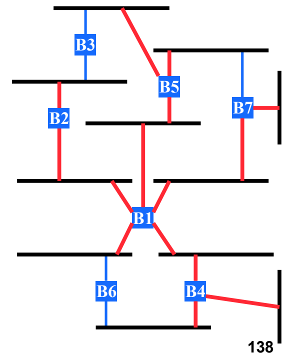
- Root of the spanning tree is the bridge with the lowest identifier.
  - All ports are part of tree
- Each bridge finds shortest path to the root.
  - Remembers port that is on the shortest path
  - Used to forward packets
- Select for each LAN the designated bridge that has the shortest path to the root.
  - Identifier as tie-breaker
  - Responsible for that LAN






# Spanning Tree Algorithm

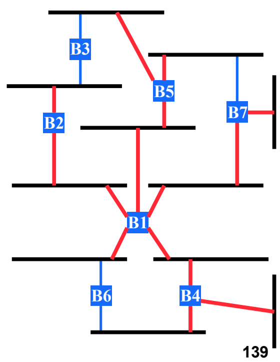
- Each node sends configuration message to all neighbors.
  - Identifier of the sender
  - Id of the presumed root
  - Distance to the presumed root
  - E.g. B5 sends (B5, B5, 0)
- When B receive a message, it decide whether the solution is better than their local solution.
  - A root with a lower identifier?
  - Same root but lower distance?
  - Same root, distance but sender has lower identifier?
- After convergence, each bridge knows the root, distance to root, root port, and designated bridge for each LAN.






# Spanning Tree Algorithm
# (part 2)

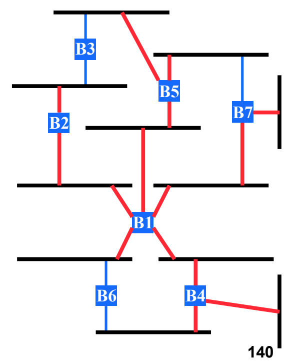
- Each bridge B can now select which of its ports make up the spanning tree:
  - B's root port
  - All ports for which B is the designated bridge on the LAN
- Bridges can not configure their ports.
  - *Forwarding state* or *blocked state*, depending on whether the port is part of the spanning tree
- Root periodically sends configuration messages and bridges forward them over LANs they are responsible for.






# Spanning Tree Algorithm
# Example

- Node B2:
  - Sends (B2, B2, 0)
  - Receives (B1, B1, 0) from B1
  - Sends (B2, B1, 1) "up"
  - Continues the forwarding forever
- Node B1:
  - Will send notifications forever
- Node B7:
  - Sends (B7, B7, 0)
  - Receives (B1, B1, 0) from B1
  - Sends (B7, B1, 1) "up" and "right"
  - Receives (B5, B5, 0) - ignored
  - Receives (B5, B1, 1) - better
  - Continues forwarding the B1 messages forever to the "right"



## Physical-Layer

## (I'm tired, use Srini's slides)

141