Course on

Computer networks

Network layer

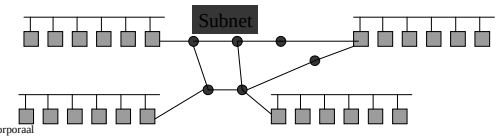
Henk Corporaal

Leiden University
2000

Network layer

Role of network layer: ***routing packets***

- deals with end-to-end communication
- aware of network topology
- choose appropriate route
- communicates with transport and data link layers



Network layer topics

- Design issues
- Routing algorithm
- Congestion control
- Internetworking
- Network layer in the Internet: IP
- Network layer in ATM Network

Network layer design issues

Services provided to the transport layer

- independent of chosen subnet technology
- shield transport layer from topology, type and number of subnets
- uniform numbering plan, across LANs and WANs

Network layer design issues

Connection-oriented or connectionless service?

- Where to put the complexity

Internet community: connectionless

- network will be unreliable anyway => handle this issue in higher layer (s)
- only SEND packet and RECEIVE packet needed
- hosts are getting powerful; so why not putting the complexity there (in the transport layer) => easy to adapt
- speed more important than accuracy

Network layer design issues

Telephone companies : NL should be connection-oriented

- Set up connection first

  Each connection has unique ID
- Negotiate parameters (quality, cost)
- Communication in both directions
- Use of Acknowledge gives automatic flow control

What happens if we run TCP via IP on ATM (fig 5-1)?

Network layer design issues

Network layer internals

- virtual circuits, versus datagrams

Virtual circuits :

- avoid routing every packet or cell independently
- all packets/cells follow same route (= session routing)
- Store # VC in routing tables (consuming some resources)

Network layer design issues

Datagrams

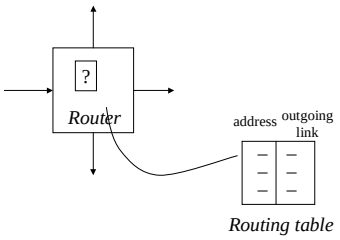
- each packet contains destination and is independently routed
- also usable for connection-oriented service (fig 5-3)

Comparison of datagram and virtual circuit services: see fig 5-2

Routing

Message / packet / cell has to pass multiple hops

Routing algorithm : decide which outgoing link to use



Routing algorithm requirements

- Correctness
- Simplicity
- Robustness: cope with changing topology
- Stability: converge quickly
- Fairness: guarantee progression (fig 5-4)
- Optimality:
  - high throughput
  - short delay
  - contradictory requirements !

Routing algorithms

Non-adaptive (static) routing:

- do not use measurements or estimates of current traffic and topology
- static routing: calculate route off-line or use deterministic algorithm
- Examples
  - Sheet-sign routing (MIMD computing)
  - E-cube routing: fixed n-dimensional array topology
  - Interval routing: arbitrary topology
  - Flooding: send packets in all directions
  - Flow based routing: can handle arbitrary changing topology

Routing algorithms

Adaptive (dynamic) routing:

- routing decision depends on network status
  - update state information regularly
  - how to get this information?
- what type of network information do we base the decision on?
  - hop count
  - distance, estimated transit time
  - traffic rate
  - congestion metric

Static routing

- Optimality criterium
- Shortest path routing
- Flooding
- Flow-based routing

Optimality of Routing

Optimality principle:

- If B on optimal path of AC than the whole path BC is part of this path

Consequence : optimal routes from all sources to a given root (= sink node) form a ***sink tree*** (fig 5-5)

⇒ no loops; each packet delivered passes limited # hops

Ideally we would like to use a sink tree for each router

Practice: routers may go down => adaptivity needed

However, sink tree routing time can be used as reference

Shortest Path Algorithm

Used as part of many routing algorithms

- build router graph G(N, L)
  - N being nodes,
  - L being the set of links (or edges)
- choose path metric (= weight of lines L)
  - # hops
  - estimated delay (using echo packets)
  - physical distance
  - queue length (of outgoing lines)

Shortest Path Algorithm

Algorithms (see e.g. 'Data networks' by Bertsekas & Gallager pg. 393)

- Dijkstra
  - complexity O (N^2)
- Bellman-Ford
  - complexity O (N^3)
  - fast if there are relatively few edges
- Floyd-Warshall
  - complexity O (N^3)
  - computes <u>all</u> shortest paths at once

Shortest Path Algorithm

Dijkstra's shortest path algorithm (fig 5-6, 5-7)

- iterate on path length

Steps to find <u>all</u> shortest paths from A

- Initially: make A permanent: P = {A} : label all neighbors j of A with Dj =d_{Aj}
- Repeat until P contains all nodes
  - find next closest node and add it to P:
    P: = P + {i} , with D_i = min D_j (j not in P)
  - Updating labels of neighbors of P
    D_j: = min [D_j, D_j + d_{ij}] (j not in P)

Key fact: the path to the next closest node i has to go through node from P only.

Flooding

Static (just like shortest path) algorithm

- Idea: forward incoming packet on all other outgoing lines

Prevent over-flooding:

- Include Hop counter HC
  - decrease HC on every Hop
  - discard packets with HC= 0
- Avoid routing the same packet twice
  - keep list of 'outstanding' packets using sequence numbers
- Selective flooding
  - route only in directions which are about right

Flooding

Properties:

- Not practical in most applications
  - generates too much traffic
- Useful only for:
  1. extremely robustness (military purpose)
  2. Broadcast
  3. Reference algorithm
  - it always finds the shortest delay path !!

Flow-based Routing

Properties

- static routing
- take network load into account !!!

Required information (in advance):

- topology
- traffic matrix F_{ij} in packets/sec flowing from i to j
  - note: this information should be known in advance, and F_{ij} should be relatively stable
- line capacity C_{ij} in kbps
- a *tentative* routing algorithm

Flow-based Routing

Steps (see fig 5-8 / 5-9) :

- Calculate traffic λ_i (frames/sec) for each line i
- Calculate expected delay T_i for end line i

$$T_i = \frac{1}{C_i / L - \lambda_i}$$

L is mean packet length (in bits)

- Calculate mean delay $T = \langle T_i / \lambda_i \rangle$

Investigate different routing protocols and choose the best one

Dynamic routing

- distance vector routing
- link state routing

Dynamic routing algorithms

Distance vector routing
  - used in early ARPANET

Each router maintains distance table

- indexed by destination number
- entry contains : outgoing link + estimated distance

How to know distance? (see fig 5-10)

- send echo packet to all neighbors; this packet gets back with timestamp ⇒ distance to neighbor
- receive regularly tables from neighbors and update own table

$$D(a,b) = \min (D(a,x) + D(x,b)) \quad \{\text{for all neighbors } x\}$$

Distance vector routing

- Count-to-infinity problem:
  - good news propagates much faster than bad news (see fig 5-11)

Many ad hoc solutions have been proposed; one of them is:

- Split Horizon hack:
  - A does not report B the distance of node X for which packets have to flow through B

Does not solve all situations (see fig 5-12)

Link State Routing

Solves slow convergence problem of the distance vector approach

Idea:

- send all nearest neighbor information to all other routers
- each routers uses this information to know the topology and determine shortest path to all nodes

Link State Routing

4 steps:

- discover neighbors (and their IDs)
- measure delay (or cost) to reach neighbours
- send link state packets to all other routers
- compute topology and shortest path to every router

Perform last three steps at

- regular intervals
- when major event happens (link / node going down or coming up)

Link State Routing

Discover neighbors:

- after booting : send HELLO packet; neighbor should respond telling its identity
  - when multiple routers are connected to LAN treat LAN as separate node (fig 5-13)

Measure delay:

- send ECHO packet; neighbors respond immediately
- calculate trip time = delay/2
- average of multiple measurements may be used
- oscillations may occur (see fig 5-14)
- alternatively use bandwidth as cost metric

Link State Routing

Sending link state packets:

- make link state packet (see fig 5-15)
- use flooding to distribute packets to all routers
- use sequence numbers to reduce traffic
  - routers keep (source, seq#) pairs and discard lower sequence numbers
- use age fields to handle corrupted sequence numbers
  - age is decremented every second
  - information is discarded when age is zero

Link State Routing

Compute new routes (when packets from all sources have been received):

- use shortest path algorithm on each node (e.g. Dijkstra's algorithm)

Problem with Link State Routing:

- large memory requirements ~ O (NK), where:
  - N: number of routers
  - K: network node degree

Link state routing used widely; e.g.:

In OSPF (open shortest path first) routing used in the Internet (see section 5.5.3)

Hierarchical Routing

Purpose: reduce routing table size and routing computation time

Solution: use multi-level topology

E.g. a 3-level network having:

k clusters
k * k regions
k * k * k nodes

Instead of requiring k^3 entries we need only 3k entries (see fig 5-17)

For N nodes optimally we need ln(N) levels

Increase in mean path length is sufficiently small

Routing for mobile hosts

Many people have mobile (and possibly wireless) computers, and want to receive email etc. on this mobile system. How to get messages there?

Idea (fig 5-18):

- each (mobile) system has home location
- each area (e.g.. LAN) has 1 home agent (HA) and 1 or more foreign agents (FAs)
- packets are rerouted from via HA to FA

Routing for mobile hosts

Steps (see fig 5-19)

- newly arrived mobile systems registers with FA
  - either mobile sends a broadcast to detect FA, or receives broadcast from some FA
- FA contacts HA of mobile system
- HA performs security checks and sends acknowledge
- HA 'tunnels' packets for mobile system
- HA sends mobile address to originator of message

Broadcast Routing

Options:

- Send lots of individual packets
  - lots of traffic
  - source needs to know all destination addresses
- Use Flooding
  - generates too many packets
- Multi-destination routing
  - packet contains list of destinations
  - router splits packet among output lines and updates destination list
  - again all destination addresses have to be known!

Broadcast Routing (cont'd)

- Use **sink tree** (= *spanning tree*)
  - not always known (e.g. not when distance vector routing is used; link state routing does have this knowledge!)
- Use reverse path forwarding (method: see next slide)
  - does not require network knowledge
  - easy to implement
  - transmission automatically finishes
  - reasonable efficient
  - no list of destinations needed
  - no special mechanism (as with flooding) needed to stop the forwarding

Reverse path forwarding

Method (fig 5-20c):

- send packet to all neighbors, but discard incoming packets from 'wrong' lines
- packet from A arriving at B on line L is 'wrong' if B does not send to A at line L
- the idea is that if a packet arrives at the wrong line it probably did not follow the shortest route

Multicast Routing

Some applications, like distributed database, regularly need multicasts: send packet to a group of nodes =>

- Group management required
  - Routers must know to which groups belong their hosts
  - Each router computes spanning tree for each supported group (fig 5-21)
  - Multicast packets are then forwarded to appropriate spanning trees only
- Problem: much memory per group needed

  #mem ~ #groups x #routers in group (in each router)

  Solution:
  - use one spanning tree per group
  - send multicast to root of this tree first

Congestion control algorithms

Problem: Network performance severely degrades at high load

Congestion Control Overview

- Introduction
- General control principles
  - open versus closed loops
- Policies for prevention
- Traffic Shaping
  - bucket algorithms
- Flow specification
- Admission control in virtual circuits subnet
- Choke packets
- Load shedding
- Jitter control
- Congestion control for multicasting

Congestion Control

At very high traffic network throughput collapses
  - see fig 5-22

Reasons:

- Mismatch between parts of system
  - router CPUs are too slow
  - lines too slow
- Sudden burst traffic
- N to 1 communication: ***Hot-spot***

Bottleneck in network: e.g. root of tree network

Result: packets get lost or get timed out

=> retransmissions => collapse of network throughput

Congestion control

What is the difference with flow control

Congestion control

- global issue (network)
- indirect feedback
  - sometimes slowdown messages are used

Flow control

- local issue (line, point-to-point connection)
- direct feedback from neighbor

General Principles of Congestion Control

In control system theory, systems are divided in:
Open Loop and Closed Loop

Open loop: no feedback; solve problems at design time (too make sure they never occur; worst case design)

Open loop issues:

- No regard to current state of network
- When to accept new traffic ?
- When to discard new traffic ?
- Make scheduling decisions

General Principles of Congestion Control

Closed loop: feedback

- Monitor the network on congestion
  - metrics: % discarded packets, queue length, # timed out packets, average delay
- Pass this information to right place
  - feedback packets: increase load
  - reserved bit field in header of every packet
  - probe packets can be sent periodically (ask about congestion)
- Adjust system operation
  - avoid oscillation (proper response time needed)
  - increase #resources
  - decrease load

Policies for prevention

Open loop policies can be applied at different layers (see fig 5-23)

Data link layer policies

- Retransmission creates extra packets
  - tune time out time
  - use selective repeat i.s.o. go back n (i.e. do not discard out of order packets)
- Acknowledgement policy
  - use piggyback
- Tight flow control (small window)

Policies for prevention

Network layer policies:

- many congestion control policies work only with virtual circuits, e.g.. reserving space
- routing
  - spread traffic
  - avoid hot spots
- discard policies
  - remove long life-time packets

Transport layer policies:

- similar as previous policies
- choose proper time-out value for sender

Traffic shaping

- Avoid bursty traffic
- Regulate the average rate
- Negotiate a contract between sender and network (ATM)
  - especially important for real-time applications

Traffic control

- Leaky bucket
- Token bucket
- Hybrid solution

Traffic control: leaky bucket

Leaky bucket algorithm (fig 5-24)

- Sender puts packets / bytes in a queue
- When queue is full the input is discarded
- Queue output data at constant packet or byte rate
- Equal to a single-server queuing system with constant service time

Implementation: use saturating byte counter

- every tick increase counter X (up till max value)
- if packet with length L comes in and X>L then X = X-L, else wait

Example : see fig 5-25 a, b

Traffic control: token bucket

- Leaky bucket algorithm too rigid
  - constant low rate
  - does not allow temporary larger output rate

Token bucket algorithm (fig 5-26)

- Bucket holds tokens, generated 1 per Δ T sec. up till certain max. value
- Idle host time builds up 'transmission rights' (= tokens)
- Packets are never discarded

Example: fig 5-25c, d, e,

Traffic control: token bucket

Calculation of burst interval S (seconds)

Given:

- C = token bucket size (bytes)
- ρ = token arrival rate (bytes/sec)
- M = burst output rate (bytes/sec)

$$C + \rho S = MS \Leftrightarrow S = C/(M - \rho)$$

Hybrid solution: leaky bucket after token buckets : fig 5-25f

Congestion Control in virtual circuits subnets

Admission control

- closed loop control
- do not allow new virtual circuits until congestion problem disappeared (compare phone call system)

Alternatives:

- route new circuits around 'problem areas' (fig 5-28)
- reserve network resources
  - always
  - or only when congestion occurs

Choke packets

Estimate congestion u by sampling certain activity f

- e.g.. Line utilization

$$u_{new} = a\, u_{old} + (1-a)\, f$$

- Give warning when u > threshold
- Send choke packet to source host when in warning state
- Original packet is tagged => it does not generate further choke packets

Problem: the host may discard choke packets

- waited fair queuing (sent/rout shorter packets first)

Hop-by-hop choke packets

Choke packets may take a long time to cause effect (see fig 5-30)

Solution:

- Choke packets should not effect the source host only, but also intermediate routers
  - routers reduce traffic in certain direction
  - more buffer space required at routers !!

Load Shedding

Allow routers to discard packets

What's the selection criterion ?

- discard old packets
  - good for video (no retransmission)
- discard new packets
  - good for file transfer (with go back n protocol)
- use packet priority
  - ATM supports a 1-bit priority field
  - users should pay for high priority

Remark: measurements show that routers should not wait too long with discarding packets

Internetworking

Deal with following problem :

- how to connect and route on networks using different protocol stacks
  - different protocols will always exist (vendor policies)
  - cheaper networks encourage variety (it becomes a departmental decision)

Interconnection of different networks is called an **internet**

(example in fig 5-33)

Internetworking

Various boxes connecting networks or segments are used:

- Physical layer: Repeaters
  - copy individual bits between cable segments
- Data link layer: Bridges
  - store and forward data link frames between LANs
- Network layer: Multi-protocol routers
  - forward packets between dissimilar networks
  - often called gateways
- Transport layer: Gateways
  - connect byte streams in the transport layer
  - e.g. TCP socket connection
- Application gateways: Proxy servers
  - firewall

Internetworking

Note that many devices combine different functionality, e.g. bridge and router functionality

However:

- bridge does not inspect network layer headers (e.g. IP headers)
- router makes decision on NL headers, like IP address

Many products are sold with wrong label

How networks differ ?

Fig 5-35 gives a number of differing characteristics

- Connectionless service may change the order of packets
- Protocol conversion: e.g.. From IP to DECnet
- Address conversion (range, hierarchy)
- Quality of service may not be provided
- Different packet length
- Different methods (if at all) for flow control, security, accounting, etc.

Concatenated Virtual Circuits

Connection is concatenation of virtual circuits over different networks (see fig 5-36)

- Gateway records #vc and builds new vc to router in next subnet
- All packets traverse same path
- works only if protocols do not differ too much
  - e.g. a reliable connection can only be guaranteed if all subnets handle reliable communication

Concatenation of virtual circuits is also common in the transport layer e.g.. a bit pipe between OSI and TCP

Connectionless internetworking

Datagram service spanning an internet (multiple networks)

- packets may travel different routes (see fig 5-37)

Problem:

- connect different network protocols: almost not possible
  - format conversion is rarely attempted
  - addresses difficult to remap

Connectionless internetworking

Advantages concatenated VC

- buffer space reservation
- guaranteed sequencing
- short headers

Advantages datagram internetworking services

- adaptive routing
- can be used over subnets that do not support virtual circuits
  - many LANs & mobile subnets do not support VCs

Tunneling

If source and destination hosts are on same type of network but there is a different network in between, *tunneling* can be used

(see fig 5-38)

- packet encapsulation performed by multi protocol router
- no packet (format) conversion needed !

Internetwork Routing

Distinguish single subnet and internetwork routing

In the latter multi protocol routers are 'talking' to each other (see fig 5-40)

- subnet routing uses interior gateway protocol
- internetwork routing uses exterior gateway protocol
- each network in an internet is called Autonomous system (AS)

Internetwork Routing

Steps for internetwork routing

1. Packet goes from host via local router and LAN to multi protocol router (on same network)
2. This router uses routing tables to choose the next multi protocol route
3. Either; forward packet (if possible) or; tunnel packet
4. Repeat steps 2 and 3 until final multi-router reached
5. Route on local destination LAN to its destination

Notes:

- countries may implement 'forced rules'
  - e.g.. A packet from Canada to Canada may not cross the border
- accounting by different countries

Fragmentation

All networks impose maximum packet payload size
- 48 bytes in ATM
- 65,515 bytes in IP

Multi protocol routers (gateways) may fragment packets

Two options (fig. 5.41):

- Transparant : packet leaves network unfragmented
  - fragments have to use same route

Fragmentation

- Non-transparant
  - fragments reassembled at destination host
  - all hosts must support this assemble feature
- Problem: if one fragment lost, end-to-end retransmission required

Better approach: choose smallest unit which every network can handle

Fire walls

Purpose: prevent leak-out and leak-in (fig 5-43)

- encryption : protects data but not connection

Firewall solution: all traffic from interior to exterior network has to pass a special host (BASTION)

Two options

- Packet filter in the network layer
  - table driven: check source and destination
- Application gateway used in proxy server
  - no direct IP traffic possible

(see e.g. www.pinewood.nl/firewall.html)

Firewalls

Network layer packet filtering characteristics:

- direct IP traffic possible
- low cost (standard IP router facility can be used)
- configuration may be nasty an error prone
- no knowledge of type of connection
  - can not disambiguate e.g.. Telnet and www (HTTP)
  - use dynamic packet filtering

Firewalls

Application layer filtering characteristics

- Not transparant for own employers
- knowledge of application protocol
  - each application requires separate proxy server
- user authentication possible
- lower flexibility
- lower performance

The Internet Network Layer: IP

Internet = collection of autonomous systems running IP

IP = Internet Protocol (provides datagram service)

Example; fig 5-44

Transport layer breaks messages up in datagrams of about 1500 bytes

Datagrams are reassembled at destination host (by Transport layer)

IP Protocol

IP datagram contains a header of minimal 20 bytes, containing fields for:

- Version (= 4 for IPv4)
- IHL: internet header length (in words ≥ 5)
- Type of Service
  - priorities for delay, throughput and reliability
  - currently hardly used by any router
- Total length
- Identification
  - label fragments belonging to same datagram

IP Protocol

- DF (do not fragment) bit
  - e.g. used for *boot-image*
  - note: datagrams < 576 bytes should be supported
- MF (more fragments) bit: is this the final fragment?
- Fragment offset: where does fragment start (in total original payload)
- TTL (time to live)
  - counter decremented at each hop
  - discard datagram when zero: send control message back to source

IP Protocol

- Protocol field:
  - indicate which transport layer protocol to use: e.g.. UDP or TCP
  - allows direct delivery to right TL protocol handler
- Header checksum
- Source and Destination IP addresses

IP addressing

Every IP-host (on the net) needs unique 32-bit address

- Five classes: A - E (fig 5-47)
- Assignment by NIC: network information centre
- Dotted decimal notation: e.g. 129.245.96.122
- Special addresses (fig 5-48):
  - 0.0.0.0 used during boot-up
  - network# = 0: own network
  - all ones: broadcast (local or on other network#)
  - 127.x.y.z : loopback for internal testing purposes

Subnets

What if company grows out of its C-class (i.e. it needs more than 256 addresses) ?

- ask for B-net: overkill
- extra C net(s)
  - worldwide change of routing tables needed
  - machines can not easily be moved between C-nets inside company

Solution: use subnets (note: overloaded term!)

E.g., split B network into 64 subnets of max. 1024 hosts each

Subnet routing

How is routing performed:

- Normally router contains table with following type of addresses
  - (network#, 0)
  - (this network#, host#)
- This is a 2-level hierarchy !
- If incoming network address not listed sent packet to default router (with more extensive table)

Subnet routing

With subnetting (for internal use only):

- Router table contains additional entries:
  - (this network#, subnet#, 0)
  - (this network#, this subnet#, host#)
- 3-level hierarchy results
- Use host masking to quickly determine subnet (see fig. 5-49)

Internet Control Protocols

Control protocols: ICMP, ARP, and RARP

1. ICMP : internet control message protocol

- report unexpected events
  - e.g.. discovered routers
- messages embedded in IP packet
- message types (see also fig 5-50):
  - destination unreachable
  - time exceeded
  - redirected : inform source that message seems to follow wrong route
  - echo: are you alive ?

Internet Control Protocols

How to map IP <=> Ethernet undresses

4 bytes 6 bytes

All Ethernet cards have unique address

- ARP: address resolution protocol: IP => Ethernet
  - host sends broadcast on LAN : give me ethernet address of W, X, Y, Z
  - address mappings are cached temporarily
  - used by almost every internet host
  - the alternative: keep configuration files up to date: far less attractive

Internet Control Protocol

Discuss example fig 5-51

5 Steps:

- Send message to e.g. name@.leidenuniv.wi.nl
- Lookup name using DNS (domain name server) DNS returns IP address
- Two options
  - local address:
    - use ARP to get ethernet address
  - remote address:
    - use ARP message; gateway responds
    - or send packet directly to gateway. Host has to know default (gateway) ethernet address.
- Intermediate (strange) network encapsulates (tunnels) packet
- On destination LAN ARP is used again to get destination ethernet address

Reverse Address Resolution Protocol

RARP: reverse of ARP: Ethernet => IP

- needed for diskless clients
  - IP address not in boot image (one image for all clients !)
- bootserver has to supply its IP address

Disadvantage: RARP frame cannot pass router !!

Solution:

- bootp : boot protocol
  - uses UDP to forward even routers
  - can supply additional info like
    - IP address of default router
    - IP address of server holding memory image
    - subnet mask to use

Routing on the internet

Distinguish:

- Interior routing: within AS (autonomous subsystem; managed by particular organization)
- Exterior routing: between ASes

Interior gateway protocol

- till 1979: distance vector protocol
  - slow convergence (count to infinity problem)
  - not usable for large systems
- since 1979: link state protocol

New standard (1990) OSPF: open shortest path first

Open Shortest path First

Requirements :

1. Open standard
2. Support variety of distance (cost) metrics
3. Dynamic routing (adaptive)
4. Support different ‘service types’
   - e.g. real-time traffic
5. load balancing
   - split load over multiple lines
6. Support hierarchical systems
   - a router cannot know the whole topology
7. Security support
8. Tunneling support

Open Shortest Path First

OSPF supports 3 types of communication

1. Point-to-point
2. Multi-access with broadcast (LANs)
3. Multi-access without broadcast (WANs)

(multi-access = can have multiple routers on network)

OSPF divides AS into areas

- An area is a generalisation of a subnet
- Outside an area its details are not visible
- Every AS has a backbone area (# 0)
  (see fig 5-53)

Open Shortest Path First

OSPF uses protocol similar to Link State Protocol

- Use ‘Hello’ packets to learn neighbours
- periodically flood link state update messages

Interior routing to other area always via backbone

Exterior Routing

BGP = Border Gateway Protocol

- implies ‘politics’
- policies are manually configured into each BGP router

Uses variant of distance vector protocol

- tables exchanged certain (distance, path) tuples
- by exchanging the whole path, the ‘count to infinity’ problem can be solved

Internet Multicasting

Multicasting uses class D addresses (224-239.x.y.z see fig 5-47)

- 28 bits for group identification
- no delivery guarantee
- permanent group addresses, e.g.. :
  - 224.0.0.1 all systems on a LAN
  - 224.0.0.2 all routers on a LAN
- temporary groups (use IGMP protocol)
  - must be created before use
  - a process can ask a host to join or leave a group
  - each minute a router asks hosts for their group (using address 224.0.0.1)
- use spanning tree to distribute information

Mobile IP

Uses solution of Sec. 5.2.8 ; using home and foreign agents

- agents regularly broadcast their services
- registration with foreign agent has time-out
- home agent use authentication before forwarding messages
- recursive tunneling
  - support for mobile systems within a mobile LAN e.g. mobile in aeroplane LAN

Classless Inter Domain Routing

Problem: Internet is running out of addresses

- C class is often too small, but B class too large

Solutions:

- use 10 bits for C class => 0.5 million C class networks
  - routing tables too large to maintain
- multi-level (i.s.o. 2-level) hierarchy
  - needs more than 32 address bits
- CIDR
  - allocate multiple adjacent C blocks
  - use Masking to quickly determine outgoing link

Classless Inter Domain Routing CDIR

Class C address spare divided into 4 parts, e.g.

- Europe gets 194.0.0.0 to 195.255.255.255 (C addresses)
- = 32 Million addresses

Masking example: assume company needs 2k addresses, and gets: 194.24.0.0 - 194.24.7.255 (8 C blocks)

- Mask = 255.255.248.0 (248 = 1111.1000b)

Routing tables contain (base address, mask) tuples
Incoming address is masked with each entry
Hit if: (address AND mask) = base address
If so, send packet to appropriate router

IPv 6 Protocol update

In near future every TV or telephone may become internet host
In 1990 work on new protocol starters with requirements:

- support billions of hosts
- reduce routing table size
- simplify protocol (make it faster
- better security (authentication and privacy)
- multicasting
- mobile hosts
- allow protocol to evolve (room for expansion)
- coexist with IPv4

IPv 6 Protocol update

Changes :

1. 16 byte addresses
2. Header simplification (fig 5-56)
   - fixed size of 40 bytes
3. Better support for options
   - some of the previously required fields are now optional
   - IHL field left out
4. Security fields
5. Priority classes
   - 0-7 non-real-time (flow controlled)
   - 8-15 real-time (no flow control)
6. Support for pseudo connections (using the flow label; experimental !)
7. Support very long datagrams (> 64k bytes)
   - for supercomputer applications

ATM

Discussion: where to put ATM

- Internet (IP) community: ATM is just a carrier for us (= IP), so it should be below the network layer
- However: ATM involves routing, therefore a network layer protocol

Network Layer in ATM

ATM protocol

- Connection oriented
- virtual circuits are unidirectional
- connection is pair of VCs with same number
- unreliable (no delivery guarantee / no acknowledgement)
- in-order arrival of cells
- two-level connection hierarchy
  - level-1: virtual path (=bundle of VCs); see fig 5-61
  - level-2: virtual connection

ATM Cell header

Two interfaces with slightly different header formats:

- UNI : user-network interface
- NNI : network-network interface

Header (see fig 5-62) fields

- GFC : (UNI only)
  - flow control between host and network
  - not really used
- VPI (virtual path ID) 8 bits
- VCI (virtual circuits ID) 16 bits

ATM Cell header

- PTI (payload type) ; see fig 5-63
  - 4 types are user supplied
  - 4 types are network supplied

  e.g., a 000 PTI may change (by network) into 010 to warn for congestion
- CLP bit (cell loss priority)
- HEC (header checksum)
  - corrects all single bit errors
  - detects 90% of multi-bit errors

ATM Connection setup

How to set up a connection? (see fig 5-64, 5-65)

- put request on VP 0, VC 5
- network returns new VC which can be used for setting up the connection
- send SETUP message to destination using new VC channel
- destination responds with CONNECT

Addresses: 20 bytes

Release: send RELEASE message: propagates to destination

ATM Routing

Standard does not specify particular routing algorithm. The idea is:

- VPI based routing between switches
- VCI based routing between switch and final host

Reason:

- Additional VC can use same VPI based routing channel ; no new table entries needed
- Smaller tables (2^{12} i.s.o. 2^{28} entries)
- Faster routing
- Quicker re-routing
  - a whole set of VCs can be re-routed by changing a single entry
- Support for private network (with company owned VIP number)

ATM Routing

A switch (router) gets requests of the form:

(source, destination, line#, VPI)

Using the destination the switch determines the outgoing line and chooses a free outgoing VPI number

This information is kept in tables (one for each outgoing link)

Example : see fig 5-67 (requests) and 5-68 (tables)

ATM Service Categories

ATM offers 4 types of service : (table 5-69)

- CBR : constant bit rate
  - used to support existing channels (like T_1)
  - real-time application may also require this service
- VBR : variable bit rate (guaranteed bandwidth)
  - real time

    average cell delay and its variation must must be tightly controlled
  - non-real time (some jitter can be tolerated)

    e.g.. multi media email

ATM Service Categories

- ABR : available bit rate
  - for bursty traffic with known average bandwidth
  - network provides congestion feedback
  - used to replace leased lines
- UBR : unspecified bit rate
  - no guarantees or promises about bandwidth
  - in case of congestion UBR cells are discarded without feedback
  - applications have to do own error and flow control

Properties of services: fig 5-70

ATM quality of service QoS

Standard defines a number of QoS parameters
For each parameter the worst case value is specified
The network is expected to meet these values
Parameters (fig 5-71)

- Peak Cell Rate : PCR
- Sustained Cell Rate : SCR (average rate)
- Minimum Cell Rate : MCR
  - if network can not guarantee this, the connection is not granted
  - Note: ABR with MCR = 0 is equal to UBR
- Cell Variation Delay Tolerance: CVDT
  - leaky bucket used to control this

ATD Quality of Service Parameters

Negotiable network Parameters

- Cell Loss Ratio = fraction of non-delivered cells
- Cell Transfer Delay = the average transmit time
- Cell Delay Variety = how uniformly cells arrive

Model used : see fig 5-72

- Cell Error Ratio : fraction of cells with wrong bits in it
- Cell Misinsertion Rate : cells / sec delivered at wrong destination

Traffic Shaping : Rate Control

Generic Cell Rate Algorithm : fig 5-73

- checks every cell according to parameters
- a cell is nonconforming if its transmission is not conform the negotiated parameters
  - discard cell
  - or : set CLP bit (mark as low priority)

Traffic Shaping : Rate Control

What happens if sender sends at higher rate than agreed PCR

- discard them, or set CLP bit (low priority cell)

Leaky bucket behavior (see also fig. 5-74)

Assume :

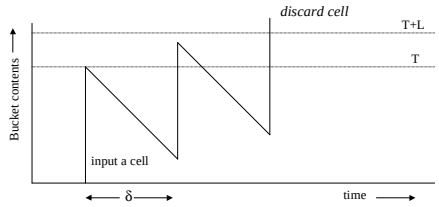
T = 1 / PCR is minimum cell inter arrival time

$\delta \leq T$ is effective cell inter arrival time

T+L = bucket cell capacity

Bucket drains at : 1 cell / T seconds

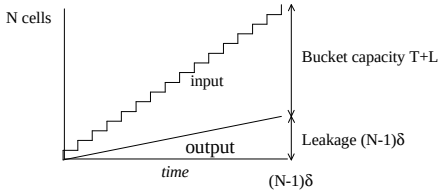
Traffic Shaping : Rate Control



Maximum allowed burst of N packets:
NT-(N-1) δ = T+L

Traffic Shaping : Rate Control

Max #cells to transmit at higher rate: $N = 1 + \frac{L}{T - \delta}$



ATM Congestion Control

Strategies :

- Admission control
- Resource reservation
  - capacity can be reserved using the SETUP message
- Rate based congestion control
  - For ARB traffic type only
  - Several proposals have been considered
  - Adopted proposal uses special Resource Management cell (RM)
    - RM cell send by sender and contains required bandwidth
    - each switch may reduce this value
    - upon return, sender knows available bandwidth
    - Incentive : if sender does not reduce transmission rate, cells may be discarded

ATM LANs

ATM networks can be used as LANs

Problem: how to provide connectionless service?

Proposal : for ATM LAN emulation (see fig 5-76)

- have separate VCs between every host pair
- use special translate server for IP => ATM address mapping
  - compare ARP protocol

Exercises

Make from chapter 5 the following exercises:

- 1, 2, 6, 8, 9, 13, 16, 20, 26, 28, 30, 33, 40, 41