



ECTE364 Data Communications

Week 6

Daniel R. Franklin

daniel@elec.uow.edu.au

35.G32, x4442

notes written in collaboration with Prof. Farzad Safaei

- The network layer is responsible for end to end transfer of data over a series of individual data networks
- It is the lowest protocol stack layer to deal with *end-to-end* communications
- Possible functions may include:
 - Connectionless (e.g. IP) or connection oriented (e.g. ATM) data transfer
 - Establishing and clearing connections, if applicable (ATM)
 - Routing, switching, addressing and flow control

Notes:

Notes:

Connectionless Mode



- Packets travel independently through the interconnected networks
- Each packet has a complete destination and source address
- Packets may be lost or can arrive out of order
- No connection setup, however each intermediate node (router) must process the complete network-layer address for each packet
- Robust in case of network failure - packets may be rerouted around failed links

ECTE364Data CommunicationsWeek 6 – p. 3

Notes: The Internet Protocol (IP) is an example of a connectionless network-layer protocol

Connection-Oriented Mode



- Packets follow a previously established path through the networks called a *virtual circuit*
- Each packet carries a *virtual circuit identifier* which determines which virtual circuit will be followed
- There is some initial delay while connection is set up, however processing delay per packet is less than for connectionless networks
- Usually provides a “reliable” service:
 - Correct packet sequencing and bounded loss can be guaranteed
- When a network node fails, all virtual circuits traversing the node are lost

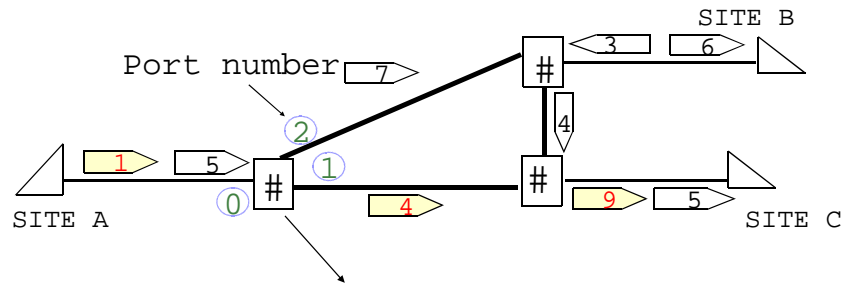
ECTE364Data CommunicationsWeek 6 – p. 4

Notes: Asynchronous Transfer Mode (ATM) is an example of a connection-oriented network layer protocol.

Some connection-oriented network-layer protocols have the capability to self-heal in the event of a node or link failure



- Based on label switching / swapping



ATM Connection Table

Input Port	Input Label	Output Port	Output Label
0	1	1	4
0	5	2	7

- For a given ‘Internetwork’ (such as the Internet), the chosen network-layer protocol must be implemented in each end system, all stations in all networks and on all *routers*

- Routers* are devices that provide connection *between* networks
- End systems (*hosts*) must have compatible protocols above the network layer to enable successful communications (e.g. TCP) - but the routers only need to understand up to the network-layer protocol

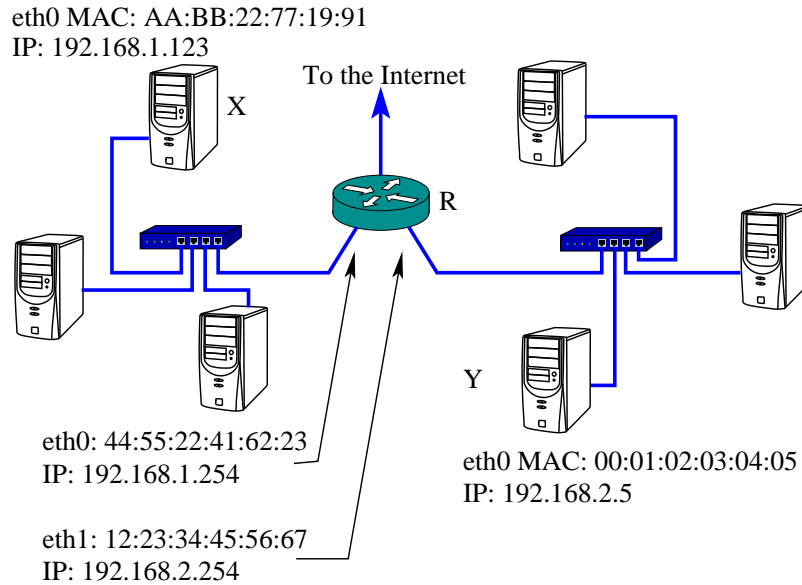
Notes:

Notes:

A Typical IP Scenario (1)



A Typical IP Scenario (2)



ECTE364Data CommunicationsWeek 6 – p. 7

Notes: In this simple scenario, two Ethernet networks, each with three hosts, are interconnected with a single router (a router is just a host with multiple network interfaces attached to different networks). How does X send data to Y?

We have not yet discussed IP in detail (this will come in the following weeks), so a few things need to be explained. In IP, every host has an IP address which is a 32-bit number. This written as four 8-bit numbers separated by a dot (.). For example, hertz.elec.uow.edu.au has the IP address 130.130.89.204. The address can be (arbitrarily) divided into two parts (bit-wise - *not* necessarily byte-wise) - a network address and a host address. Depending on the size of the network, the network part of the address may require more or less bits. For example, on the main SECTE lab subnet, we have the network address 130.130.92.0/24, which means the first 24 bits (130.130.92) are the network address and the last 8 bits (which could be 0-255) identify the specific host. For the example provided here, we assume that no more than 255 hosts are within a given network - the first network address is 192.168.1.0/24 and the second 192.168.2.0/24.

- In transferring a block of data from station X to Y :
 - X 's network layer receives blocks of data to send to Y from the transport layer
 - The IP layer attaches a header containing the global (network-layer) address of Y (in this case, IP address 192.168.2.5) as destination, and the network-layer address of X as source (IP address 192.168.1.123)
 - The whole block (payload + network-layer header) is now called a *datagram*

ECTE364Data CommunicationsWeek 6 – p. 8

Notes:



- X 's network-layer implementation recognizes that destination Y is not on X 's network - it does this by examining its own network settings and the destination address
- Therefore, it is necessary to send the datagram to a *router*. In our network there is only one router - it is called the *default gateway*.
- However, in order to do this, the network-layer datagram must be encapsulated in a data-link-layer frame for transmission in the local network.
- What if X doesn't know the data-link-layer (MAC) address of the router R ?

ECTE364Data CommunicationsWeek 6 – p. 9

Notes: The network configuration of each host specifies the address and a 'net-mask' - which is 32 bits long, and consists of a block of ones followed by a block of zeros (for example, 255.255.255.0). To see if a destination is on the same network as the sending host, a bitwise AND is performed between the mask and destination address. If the result is the same as when we AND the hosts own IP address and the netmask, the destination must be on the same network. Therefore, there is no need to forward to the gateway (router) - we can deliver the datagram locally through the local switch.

- An Address Resolution Protocol (ARP) is needed in order to identify the MAC address associated with an IP address
- A *broadcast* is sent by X to all machines in the local network, asking “**Who has** network-layer address corresponding to R 's IP address 192.168.1.254?”
- This ARP request is an Ethernet frame with
 - Source address equal to X 's MAC address (AA:BB:22:77:19:91)
 - Destination set to the broadcast address (FF:FF:FF:FF:FF:FF).

ECTE364Data CommunicationsWeek 6 – p. 10

Notes: Address resolution is used (as opposed to manually specifying the gateway MAC address in every host on the network) since the gateway may *change* (hardware upgrade, maintenance, replacement of failed hardware etc.).

Caches time out every few minutes, to allow for changes in hosts (e.g. moved, turned off, etc.). All hosts update their caches when a new (i.e. previously unseen) ARP response passes.



- *Only R* sends back an ARP reply to *X*, with
 - Source field of the Ethernet frame set to *R*'s MAC address
 - Destination equal to *X*'s MAC address
- *R* and *X* cache the IP:MAC mappings, retaining the information for a few minutes
- Once the ARP lookup is complete, *X* forwards the datagram to *Y*, encapsulated in an Ethernet frame with
 - Source set to *X*'s MAC address
 - Destination set to *R*'s MAC address

Notes:

- The router receives the frame, removes the IP datagram and inspects the IP header. There are three possibilities:
 1. The destination station *Y* is connected directly to one of the subnetworks to which the router is attached - therefore, local delivery is possible;
 2. To reach the destination one or more additional routers must be traversed, and we have an explicit rule to specify which one we should forward the datagram to; or
 3. We don't have an explicit rule set - forward to the router's default gateway.

Notes: As before, this routing decision is based on the destination address; a bitwise AND is performed between the destination address and the netmask of all networks to which the router is attached; the results are compared with the bitwise AND between the router's IP address on each network with the netmask. If there is a match, the router can deliver the datagram to the destination in an Ethernet frame over that network interface. If not, it should send the datagram to the default gateway (or to a specific gateway if it knows something about how to get the packet to its destination - more on this later!).

There is also another possibility - we have no way to deliver the packet - for example, if the destination is on another network that we cannot reach either directly or indirectly. In this case the packet will be dropped.



- So far we have introduced
 - A mechanism for resolving IP addresses to lower layer interface identifiers
 - A mechanism for encapsulation of IP datagrams over the underlying link layer protocols
- A functional network (inter-networking) layer also requires
 - A global addressing system (we will discuss IP's addressing scheme in detail in the next lecture)
 - A mechanism for finding paths through networks (routing)

Notes:

- The aim of routing is to determine a path from source to destination
 - In a connection oriented (or virtual circuit) network, the path is determined during the connection setup
 - In a connectionless (datagram) network, the path is determined independently for each packet
- Packets will often require multiple “hops” to reach their intended destination.
- Generally, more than one route is available

Notes: The word “route” is pronounced the US-English way here (“rowte” rather than “root”).



- A mechanism is needed to discover an appropriate path from the source to an arbitrary destination
 - Each host and router holds a table which associates networks with particular routers
 - An entry in the table essentially says 'to get to network N, forward packets to router R'
 - This table is used to make routing decisions
 - For each incoming packet, the host or router looks at the destination address and iterates through the entries in the table until it finds a match
 - If no match is found, the packet is sent to a 'default gateway' router

ECTE364Data CommunicationsWeek 6 – p. 15

Notes:

- The term *decision place* refers to which node or nodes in a network are responsible for the routing decision
- Routing decisions can be made:
 - At each node (distributed)
 - At a central node (centralized)
 - At the originating node (source based)
- The decision time and decision place are independent design variables

ECTE364Data CommunicationsWeek 6 – p. 16

Notes:



- Routing decisions are based on minimising a cost measure or metric (often also referred to as “distance”)
- This cost measure can be in terms of one or more of:
 - Number of hops;
 - Delay;
 - Throughput; and
 - Cost.
- The more information that is available and the more frequently it is updated, the more likely that the best routing decision will be made.

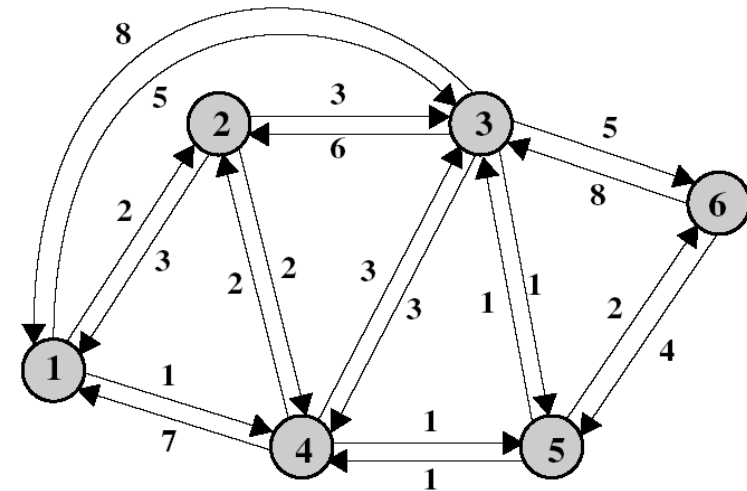
Notes:

- Over time, a large number of different strategies have evolved. These include:
 - Static (fixed) routing - appropriate for small networks, networks with a limited number of network interfaces and gateways
 - Flooding - often used in highly dynamic networks (e.g. mesh networks)
 - Random - as for flooding, but more efficient
 - Dynamic configuration - routers communicate with each other and share information which can be used to find good routes through the network

Notes: Notes: An example of a dynamic network topology is a wireless mesh network - nodes and routers may move around and be switched on and off at any time.



- The simplest approach - each host and router has a single, permanent route for each source-destination pair of nodes in the network
- Routes determined based on a least cost criterion (e.g. minimise delay)
- Routes are *fixed* - they do not adapt to changing network conditions
- A central routing directory is usually created based on the least-cost criterion.
- From it, forwarding tables can be developed to be placed in routers.



Notes:

Notes:



From Node

	1	2	3	4	5	6
1	-	1	2	4	4	5
2	2	-	5	3	3	5
3	4	3	-	5	4	5
4	4	4	5	-	4	5
5	4	4	5	5	-	5
6	4	4	5	5	6	-

To Node

Destination(1)	Next Node
2	2
3	4
4	4
5	4
6	4

Destination(2)	Next Node
1	1
3	3
4	4
5	4
6	4

Destination(3)	Next Node
1	5
2	5
4	5
5	5
6	5

Destination(4)	Next Node
1	2
2	2
3	5
5	5
6	5

Destination(5)	Next Node
1	4
2	4
3	3
4	4
6	6

Destination(6)	Next Node
1	5
2	5
3	5
4	5
5	5

Note that each routing table shows only the next node to the destination (and not the complete route)

Notes:

Notes:

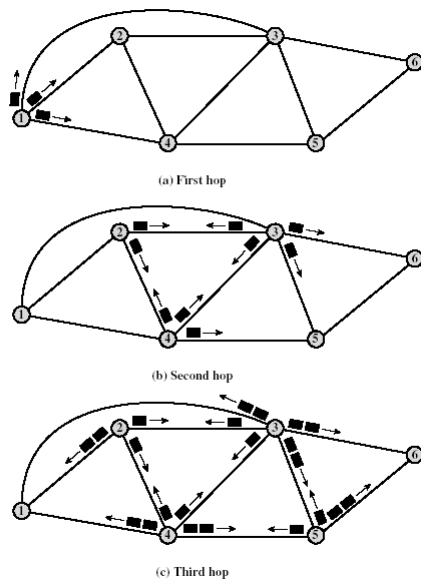


- The routes at each node are developed from the central directory and distributed
- Simple but inflexible; the strategy cannot react to changes easily, requires manual configuration and maintenance
- There is no difference between the datagram and virtual circuit routing with static routes
- All packets from a given source will always traverse the same route

Notes:

- This technique requires no routing table at all - nor any prior knowledge of the network structure!
- Any packet sent by a host is broadcast to all directly-connected nodes
- Incoming packets are retransmitted on every link except the incoming link
- Eventually one or more copies will arrive at the destination
- Each packet is uniquely numbered - so duplicates can be discarded
- Nodes can remember packets already forwarded to keep network load in bounds

Notes: Flooding is good in ad-hoc or mesh networks, since the topology is always changing. Any centralised strategy will perform poorly in such an environment.



- Advantages of flooding:
 - Robust (every possible route is tried)
 - Minimum hop route is always used (but so are many others)
 - Every node is visited (this property is often used in routing strategies to send information to all nodes)
- Disadvantages of flooding:
 - Results in high traffic load

Notes:

Notes:



- This method has the simplicity and robustness of flooding techniques
- Far less traffic is generated
- The node selects only one outgoing path for retransmission of an incoming packet
- A node may utilize links in a round-robin fashion, or apply a cost factor
- No routing table is required - no prior knowledge of the network is needed either
- Route is typically not least cost nor minimum hop

Notes: Random routing is a compromise between routing efficiency and network utilisation.



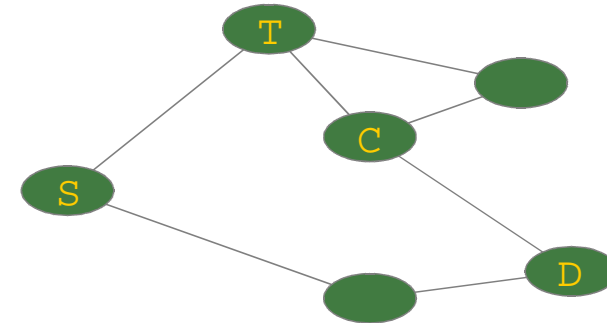
- Most packet networks use adaptive routing strategies
- Routing decisions are not fixed and can adapt to changing network conditions, such as
 - Failure at node or link
 - Time-varying congestion
- Requires a mechanism for obtaining information about the network structure
- Least-cost routes are calculated using routing optimisation algorithms

Notes: Adaptive routing incurs a greater computational cost, but offers higher efficiency than flooding/random routing. However, routing information needs to be distributed through the network (data overhead) and changes in network topology (link failures etc.) will not immediately register with all routers. This delay may cause some packet loss when the network is disturbed.



- Adaptive routing algorithms require 3 steps:
 1. Nodes collect information about link status (neighbor reachability and the cost of each available link)
 2. Nodes exchange this information with each other
 3. Nodes run a least-cost algorithm based on the gathered information to determine new routes
- The process can be periodically repeated to take into account new network conditions (congestion etc.)

- Theorem: If the shortest path from S to D is concatenation of ST and TD paths, then the shortest path from T to D is also TD



Notes: Of course, by altering the routing table, traffic flows through the network will also change. This affects congestion levels - so using congestion as a cost metric often results in an unstable network.

Notes:



- The previous theorem assumes that all network nodes are looking at the same network
- During the transition period when the network topology changes this may not be the case
 - Hence routing loops may occur
- The period of transition when routing inconsistency may occur is called the convergence period

- Problem: Given a cost assigned to each link between two nodes X and Y , find the least-cost path
- Two common approaches for adaptive distributed routing are:
 - Distance Vector Routing (based on Bellman-Ford algorithm)
 - Link State Routing (based on Dijkstra's algorithm)
- These algorithms achieve the same aim but differ in terms of performance and complexity



- Distance Vector Routing is based on the Bellman-Ford algorithm
- Each node maintains a set of minimum distances between it and all other nodes in the network (initially distances are set to infinity apart from links to immediate neighbors)
- Nodes exchange this information with immediate neighbors
- When a node x receives an update from node y , it compares its current minimum-distance vectors with the sum of the distance to node y and the distances listed in y 's distance vector

ECTE364Data CommunicationsWeek 6 – p. 33

Notes:

- If a shorter path exists, x replaces the old entry in its distance vector
- If any elements have changed, the new distance vector is distributed to all one-hop neighbors
- Distance vector routing works well, but suffers from a large delay when routing updates need to propagate over many hops
- Widely-used implementations within large intranets include RIP and IGRP, while EGP and BGP employ some DV concepts and are used to manage routes between large organisations on the Internet

ECTE364Data CommunicationsWeek 6 – p. 34

Notes: BGP is the most widely used inter-organisation routing protocol. However, it is extremely complex as its underlying algorithm also needs to take into account many rules which are imposed due to business relationships between core network providers and their customers.



- Link State Routing is also a distributed routing protocol in which each node n employs *flooding* to distribute information about the cost of all links to which it is connected
- The algorithm allows all nodes in the network to quickly build up a map of the network, with the associated costs
 - Each node then performs a complete computation of the best routes from itself to all other nodes
- Uses the “shortest path first” Dijkstra algorithm
 - The result of computation will be the next hop for each entry in the forwarding table

ECTE364Data CommunicationsWeek 6 – p. 35

Notes: The general computational complexity of Dijkstra's algorithm, in a network with N nodes and M edges, is $O(N^2)$. If $M \ll N^2$, it can be implemented much more efficiently (in $O(M + N \log N)$). The Bellman-Ford algorithm, by contrast, has a performance of between $O(M)$ and $O(MN)$.

- The OSPF protocol implements link-state routing and runs directly on top of the Internet's network layer. It has three parts:
 - The “hello protocol” used to check if links are up - sends hello packets every “hello-interval” seconds
- A change of status is distributed to all nodes using a “flooding protocol”
 - The node detecting the change will send an update to its immediate neighbours who in turn relay to others
- Initial synchronisation of router databases is performed through the “exchange” protocol

ECTE364Data CommunicationsWeek 6 – p. 36

Notes: Two adjacent routers exchange the description of their databases using “database description packets”



- Stated as follows:
 - Find the shortest paths from a given source node to all other nodes by developing paths in order of increasing path length
 - The algorithm proceeds in stages...
 - By the k th stage, the shortest path to the k nodes closest to the source node have been determined

- s = source node
- $w(i, j)$ = link cost from node i to node j
- $w(i, j)$ is set to infinity if the link (i, j) does not exist
- T = set of nodes incorporated into algorithm
- $L(n)$ = cost of the tentative least cost path from node n to node s

Notes:

Notes:



- Initialize:
 - $T = \{s\}; L(s) = 0; L(n) = w(s, n)$ for $n \neq s$

- Find a node v outside T such that:

$$L(v) = \min_{z \notin T} L(z)$$

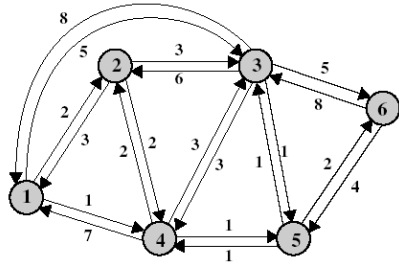
- In other words, a node outside T that has the least cost path from node s
- Incorporate v into T .

- Update all the tentative least cost paths $L(n)$ with the new node v included
 - That is, update $L(n)$ to be equal to $\min\{L(Z), L(v) + w(v, z)\}$
- Ends when all nodes added to (set) T or until $L(z) = \infty$ for every node outside T
- At the end the set of $L(n)$ is the shortest distance from node n to s

Dijkstra's Algorithm



Notes:



References

- [1] James F. Kurose and Keith W. Ross. *Computer Networking: A Top-Down Approach*, chapter 4.1, 4.2, 4.3, 4.4 4.5. Addison-Wesley, 4th edition, 2008.

<i>Itm</i>	<i>T</i>	<i>L(2)</i>	Path	<i>L(3)</i>	Path	<i>L(4)</i>	Path	<i>L(5)</i>	Path	<i>L(6)</i>	Path
1	{1}	2	1-2	5	1-3	1	1-4	∞	-	∞	-
2	{1,4}	2	1-2	4	1-4-3	1	1-4	2	1-4-5	∞	-
3	{1,2,4}	2	1-2	4	1-4-3	1	1-4	2	1-4-5	∞	-
4	{1,2,4,5}	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6
5	{1,2,3,4,5}	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6
6	{1,2,3,4,5,6}	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6