Reti di Calcolatori
AA 2011/2012

UNIVERSITÀ DEGLI STUDI DI TRENTO

http://disi.unitn.it/locigno/index.php/teaching-duties/computer-networks

Il livello Rete
Il Protocollo IP
Internet Routing

Renato Lo Cigno

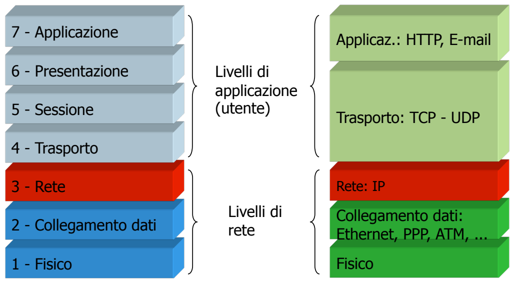
Acknowledgement

- *Credits*
  - *Part of the material is based on slides provided by the following authors*
    - *Jim Kurose, Keith Ross, "Computer Networking: A Top Down Approach," 4th edition, Addison-Wesley, July 2007*
    - *Douglas Comer, "Computer Networks and Internets," 5th edition, Prentice Hall*
    - *Behrouz A. Forouzan, Sophia Chung Fegan, "TCP/IP Protocol Suite," McGraw-Hill*, January 2005
- La traduzione è in generale opera (e responsabilità) del docente

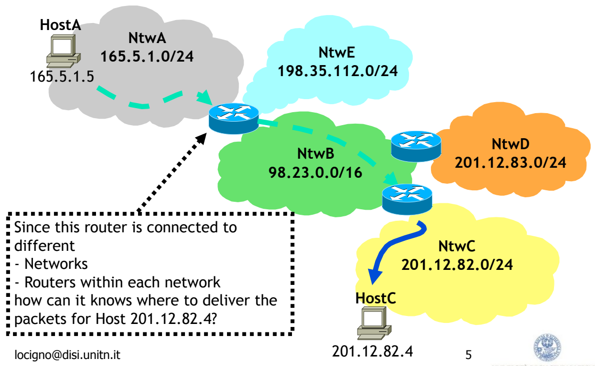
Contenuto e temi

- Spazio di indirizzamento
- Indirizzi IP e loro uso
- Configurazione dei PC e delle reti
- Consegna dei pacchetti
- Instradamento e Routing

Livello Rete

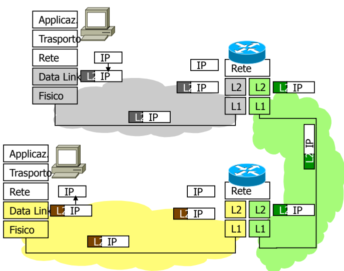


Consegna Diretta e Indiretta



Prospettiva globale

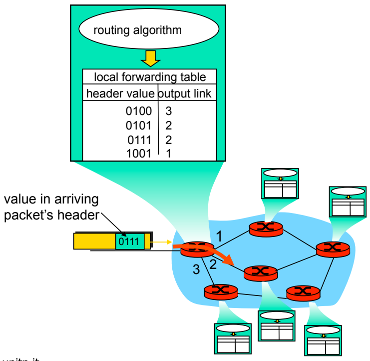
- Trasporto dei pacchetti da sorgente a ricevitore. I pacchetti contengono un segmento di livello trasporto
- I pacchetti sono incapsulati in trame L2
- Al ricevitore i segmenti sono estratti dai pacchetti e consegnati al livello trasporto
- I protocolli di rete sono in tutti gli host e router
- Un router deve esaminare l'intestazione di tutti i pacchetti che lo attraversano



Two Key Network-Layer Functions

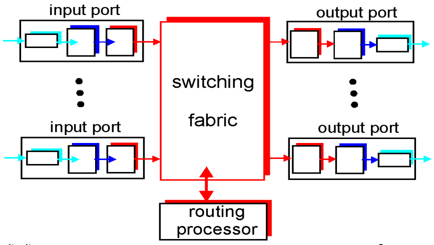
- Instradamento (Routing)
  - Trovare il percorso dalla sorgente alla destinazione
  - ➔ Algoritmi di Routing
    - Simile a pianificare un viaggio: devo determinare le strade da fare e gli incroci in cui cambiare la mia strada
- Inoltro (Forwarding)
  - Funzione che esegue il trasporto dei pacchetti dagli ingressi alle uscite dei router ... dato che il percorso è già noto
    - Simile a prendere l'uscita giusta di una rotonda, sapendo che devo andare in una specifica direzione
- Entrambe queste funzioni richiedono uno spazio di indirizzamento appropriato ... e i relativi indirizzi

Relazione tra routing e forwarding



Router Architecture Overview

- Two key router functions:
  - run routing algorithms/protocol (RIP, OSPF, BGP)
  - forwarding datagrams from incoming to outgoing link



IL PROTOCOLLO IP

The IP Datagram

- TCP/IP uses the name IP datagram to refer to a packet
- Each datagram consists of a header
  - 20 to 60 bytes in length and contains information essential to routing and delivery
- followed by data area (payload)
  - The amount of data carried in a datagram is not fixed
  - The size of a datagram is determined by the application that sends data
  - A datagram can contain as little as a single octet of data or at most 64K

The IP Datagram Header Format

- What does a datagram header contain?
  - It contains information used to forward the datagram
- A datagram head contains information, such as:
  - the address of the source (the original sender)
  - the address of the destination (the ultimate recipient)
  - and a field that specifies the type of data being carried in the payload
- Each address in the header is an IP address
  - MAC addresses for the sender and recipient do not appear
- Each field in an IP datagram header has a fixed size
  - which makes header processing efficient

The IP Datagram Header Format

0	4	8	16	19	24	31
VERS	H. LEN	SERVICE TYPE	TOTAL LENGTH			
IDENTIFICATION			FLAGS	FRAGMENT OFFSET		
TIME TO LIVE		TYPE	HEADER CHECKSUM			
SOURCE IP ADDRESS						
DESTINATION IP ADDRESS						
IP OPTIONS (MAY BE OMITTED)					PADDING	
BEGINNING OF PAYLOAD (DATA BEING SENT) ⋮						

The IP Datagram Header Format

- VERS
  - Each datagram begins with a 4-bit protocol version number
- H.LEN
  - 4-bit header specifies the number of 32-bit quantities in the header
  - If no options are present, the value is 5
- SERVICE TYPE
  - 8-bit field that carries a class of service for the datagram
    - seldom used in practice
- TOTAL LENGTH
  - 16-bit integer that specifies the total number of bytes in the datagram
    - including both the header and the data

The IP Datagram Header Format

- IDENTIFICATION
  - 16-bit number (usually sequential) assigned to the datagram
    - used to gather all fragments for reassembly to the datagram
- FLAGS
  - 3-bit field with individual bits specifying whether the datagram is a fragment
    - If so, then whether the fragment corresponds to the rightmost piece of the original datagram
- FRAGMENT OFFSET
  - 13-bit field that specifies where in the original datagram the data in this fragment belongs
  - the value of the field is multiplied by 8 to obtain an offset

The IP Datagram Header Format

- TIME TO LIVE
  - 8-bit integer initialized by the original sender
  - it is decremented by each router that processes the datagram
  - if the value reaches zero (0)
    - the datagram is discarded and an error message is sent back to the source
- TYPE
  - 8-bit field that specifies the type of the payload
- HEADER CHECKSUM
  - 16-bit ones-complement checksum of header fields
- SOURCE IP ADDRESS
  - 32-bit Internet address of the original sender

The IP Datagram Header Format

- DESTINATION IP ADDRESS
  - The 32-bit Internet address of the ultimate destination
- IP OPTIONS
  - Optional header fields used to control routing and datagram processing
  - Most datagrams do not contain any options
- PADDING
  - If options do not end on a 32-bit boundary
    - zero bits of padding are added to make the header a multiple of 32 bits

FRAMMENTAZIONE DEI PACCHETTI IP

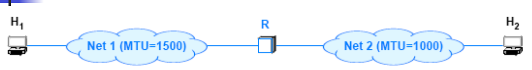
MTU and Datagram Fragmentation

- Each hardware technology specifies the maximum amount of data that a frame can carry
  - The limit is known as a Maximum Transmission Unit (MTU)
- Network hardware is not designed to accept or transfer frames that carry more data than the MTU allows
  - A datagram must be smaller or equal to the network MTU
    - or it cannot be encapsulated for transmission
- In an internet that contains heterogeneous networks, MTU restrictions create a problem
- A router can connect networks with different MTU values
  - a datagram that a router receives over one network can be too large to send over another network

MTUs for some networks

Protocol	*MTU*
Hyperchannel	65,535
Token Ring (16 Mbps)	17,914
Token Ring (4 Mbps)	4,464
FDDI	4,352
Ethernet	1,500
X.25	576
PPP	variabile

MTU and Datagram Fragmentation



- Example: a router interconnects two networks with MTU values of 1500 and 1000
  - Host H_1 attaches to a network with an MTU of 1500
    - and can send a datagram that is up to 1500 octets
  - Host H_2 attaches to a network that has an MTU of 1000
    - which means that it cannot send/receive a datagram larger than 1000 octets
  - If host H_1 sends a 1500-octet datagram to host H_2
    - router R will not be able to encapsulate it for transmission across network 2

MTU and Datagram Fragmentation

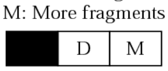
- When a datagram is larger than the MTU of the network over which it must be sent
  - the router divides the datagram into smaller pieces called fragments
  - and sends each fragment independently
- A fragment has the same format as other datagrams
  - a bit in the FLAGS field of the header indicates whether a datagram is a fragment or a complete datagram
- Other fields in the header are assigned information for the ultimate destination to reassemble fragments
  - to reproduce the original datagram
- The FRAGMENT OFFSET specifies where in the original datagram the fragment belongs

MTU and Datagram Fragmentation

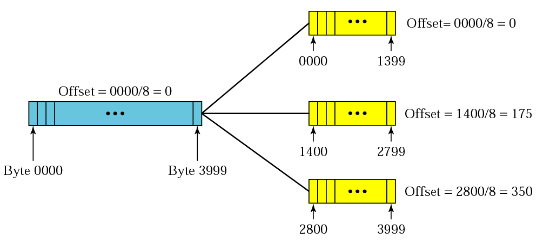
- A router uses the network MTU and the header size to calculate
  - the maximum amount of data that can be sent in each fragment
  - and the number of fragments that will be needed
- The router then creates the fragments
  - It uses fields from the original header to create a fragment header
    - For example, the router copies the IP SOURCE and IP DESTINATION fields from the datagram into the fragment header
  - It copies the appropriate data from the original datagram into the fragment
  - Then it transmits the result

Flags field

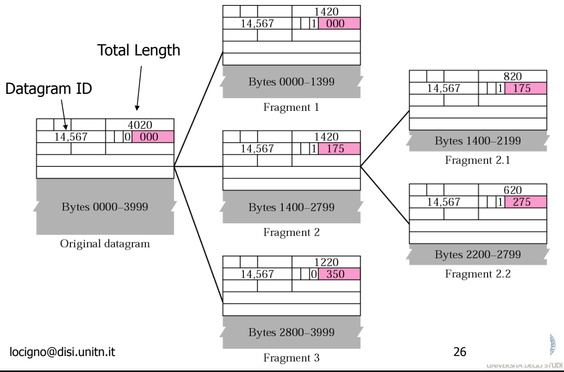
D: Do not fragment
M: More fragments



Fragmentation example



Fragmentation example



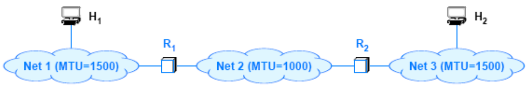
Questions

- A packet has arrived with an M bit value of 0. Is this the first fragment, the last fragment, or a middle fragment? Do we know if the packet was fragmented?
  - If the M bit is 0, it means that there are no more fragments; the fragment is the last one. However, we cannot say if the original packet was fragmented or not. A nonfragmented packet is considered the last fragment
- A packet has arrived with an M bit value of 1. Is this the first fragment, the last fragment, or a middle fragment? Do we know if the packet was fragmented?
  - If the M bit is 1, it means that there is at least one more fragment. This fragment can be the first one or a middle one, but not the last one. We don't know if it is the first one or a middle one; we need more information (the value of the fragmentation offset). See also the next example.

Questions

- A packet has arrived with an M bit value of 1 and a fragmentation offset value of zero. Is this the first fragment, the last fragment, or a middle fragment?
  - Because the M bit is 1, it is either the first fragment or a middle one. Because the offset value is 0, it is the first fragment
- A packet has arrived in which the offset value is 100. What is the number of the first byte? Do we know the number of the last byte?
  - To find the number of the first byte, we multiply the offset value by 8. This means that the first byte number is 800. We cannot determine the number of the last byte unless we know the length of the data.

Reassembly of a Datagram from Fragments



- Example: packets sent from H1 to H2
  - if host H1 sends a 1500-octet datagram to host H2, router R1 will divide the datagram into two fragments, which it will forward to R2
  - Router R2 does not reassemble the fragments
    - Instead R uses the destination address in a fragment to forward the fragment as usual
  - The ultimate destination host, H2, collects the fragments and reassembles them to produce the original datagram

Reassembly of a Datagram from Fragments

- Requiring the ultimate destination to reassemble fragments has two advantages:
  - It reduces the amount of state information in routers
    - When forwarding a datagram, a router does not need to know whether the datagram is a fragment
  - It allows routes to change dynamically
    - If an intermediate router were to reassemble fragments, all fragments would need to reach the router
- By postponing reassembly until the ultimate destination
  - IP is free to pass some fragments from a datagram along different routes than other fragments

The Consequence of Fragment Loss

- A datagram cannot be reassembled until all fragments arrive
- The receiver must save (buffer) the fragments
  - In case missing fragments are only delayed
  - A receiver cannot hold fragments an arbitrarily long time
- IP specifies a maximum time to hold fragments
- When the first fragment arrives from a given datagram
  - the receiver starts a reassembly timer
- If all fragments of a datagram arrive before the timer expires
  - the receiver cancels the timer and reassembles the datagram
- Otherwise the receiver discards the fragments

The Consequence of Fragment Loss

- The result of IP's reassembly timer is all-or-nothing:
  - either all fragments arrive and IP reassembles the datagram,
  - If not then IP discards the incomplete datagram
- There is no mechanism for a receiver to tell the sender which fragments have arrived
  - The sender does not know about fragmentation
- If a sender retransmits, the datagram routes may be different
  - a retransmission would not necessarily traverse the same routers
    - also, there is no guarantee that a retransmitted datagram would be fragmented in the same way as the original

GLI INDIRIZZI DI INTERNET: IPV4

Gli indirizzi di Internet

- Lo spazio di indirizzamento è un componente critico di Internet
- Tutti gli host e i router devono usare uno schema di indirizzamento **uniforme**
- Gli indirizzi Unicast, che identificano una specifica interfaccia devono essere **unici**
- Esistono due spazi di indirizzamento specificati per Internet
  - **IPv4**: quello attualmente in uso con indirizzi a 32 bit
  - IPv6: il sistema di indirizzamento che avrebbe dovuto sostituire IPv4, ma che continua a non farlo
    - indirizzi a 128 bit
    - funzioni “avanzate”
    - esistono molte “isole” IPv6 e ormai tutti i router dei maggiori vendor lo supportano

The IP Addressing Scheme

- MAC addresses do not suffice because
  - the Internet can include multiple network technologies
  - and each technology defines its own MAC addresses
- IP addresses are supplied by protocol software
  - They are not part of the underlying network
- Each host is assigned a unique 32-bit number
  - known as the host's IP address or Internet address
- When sending a packet across the Internet, sender’s protocol software must specify
  - its own 32-bit IP address (the source address)
  - and the address of the intended recipient (the destination address)

Dotted Decimal Notation

- Instead of writing 32 bits, a notation more convenient for humans to understand is used
- Notation, known as dotted decimal notation, is
  - express each 8-bit section of a 32-bit number as a decimal value
  - use periods to separate the sections
- Dotted decimal treats each octet (byte) as an unsigned binary integer
  - the smallest value, 0
    - occurs when all bits of an octet are zero (0)
  - the largest value, 255
    - occurs when all bits of an octet are one (1)
  - dotted decimal addresses range

    0.0.0.0 through 255.255.255.255

Dotted Decimal Notation: examples

32-bit Binary Number	Equivalent Dotted Decimal
10000001 00110100 00000110 00000000	129 . 52 . 6 . 0
11000000 00000101 00110000 00000011	192 . 5 . 48 . 3
00001010 00000010 00000000 00100101	10 . 2 . 0 . 37
10000000 00001010 00000010 00000011	128 . 10 . 2 . 3
10000000 10000000 11111111 00000000	128 . 128 . 255 . 0

The IP Address Hierarchy

- IP address is divided into two parts:
- A prefix ➔ identifies the physical network to which the host is attached (also known ad NetID)
  - Each network in the Internet is assigned a unique network number
- A suffix ➔ identifies a specific computer (host/node) on the network (also known ad HostID)
  - Each computer on a given network is assigned a unique suffix
- IP address scheme guarantees two properties:
  - Each computer is assigned a unique address
  - Network number (prefix) assignments must be coordinated globally
    - Suffixes are assigned locally without global coordination

INDIRIZZAMENTO CON CLASSI (OBSOLETO)

Schema di organizzazione degli indirizzi usato fino alla metà degli anni '90 e basato su una divisione statica tra NetID e HostID

È uso ancora oggi riferire l'organizzazione degli indirizzi ad un concetto (e terminologia) di classe

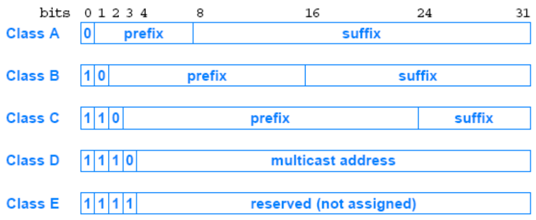
Classes of IP Addresses: bit tradeoff

- How many bits to place in each part of an IP address?
  - The prefix needs sufficient bits to allow a unique network number to be assigned to each physical network in the Internet
  - The suffix needs sufficient bits to permit each computer attached to a network to be assigned a unique suffix
- No simple choice was possible to allocate bits!
  - Choosing a large prefix accommodates many networks
    - but limits the size of each network
  - Choosing a large suffix means each physical network can contain many computers
    - but limits the total number of networks

Original Classes of IP Addresses

- Internet contains a few large physical networks and many small networks
  - the designers chose an addressing scheme to accommodate a combination of large and small networks
- The original classful IP addressing divided the IP address space into 3 primary classes
  - each class has a different size prefix and suffix
- The first four bits of an IP address determined the class to which the address belonged
  - It specifies how the remainder of the address was divided into prefix and suffix

Original Classes of IP Addresses



Division of the Address Space

- The classful scheme divided the address space into unequal sizes
- The designers chose an unequal division to accommodate a variety of scenarios
  - For example, although it is limited to 128 networks, class A contains half of all addresses
    - The motivation was to allow major ISPs to each deploy a large network that connected millions of computers
  - Similarly, the motivation for class C was to allow an organization to have a few computers connected on a LAN

Division of the Address Space

Address Class	Bits In Prefix	Maximum Number of Networks	Bits In Suffix	Maximum Number Of Hosts Per Network
A	7	128	24	16777216
B	14	16384	16	65536
C	21	2097152	8	256

Authority for Addresses

- Internet Corporation for Assigned Names and Numbers (ICANN) authority has been established
  - to handle address assignment and adjudicate disputes
- ICANN does not assign individual prefixes
  - Instead, ICANN authorizes a set of registrars to do so
- Registrars make blocks of addresses available to ISPs
  - ISPs provide addresses to subscribers
- To obtain a prefix
  - a corporation usually contacts an ISP

INDIRIZZAMENTO SENZA CLASSI E CIDR

Schema in uso attuale con divisione dinamica tra NetID e HostID

CIDR (Classless Inter-Domain Routing) consente l'instradamento globale senza usare la nozione di classe

Subnets and Classless Addressing

- As the Internet grew the original classful addressing scheme became a limitation
- Everyone demanded a class A or class B address
  - So they would have enough addresses for future growth
    - but many addresses in class A and B were unused
- Two mechanisms, closely related, were designed to overcome the limitation
  - Subnet addressing
  - Classless addressing
- Instead of having three distinct address classes, allow the division between prefix/suffix on an arbitrary bit boundary

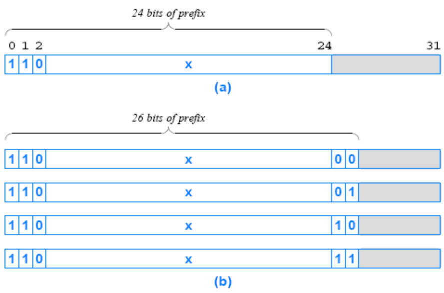
Classless Addressing: Motivation

- Consider an ISP that hands out prefixes. Suppose a customer of the ISP requests a prefix for a network that contains 55 hosts
  - classful addressing requires a complete class C prefix
  - only 6 bits of suffix are needed to represent all possible host values
    - means 190 of the 254 possible suffixes would never be assigned
  - most of the class C address space is wasted
- For the above example
  - classless addressing allows the ISP to assign
    - a prefix that is 26 bits long
    - a suffix that is 6 bits long

Classless Addressing: Example

- Assume an ISP owns a class C prefix
  - Classful addressing assigns the entire prefix to one organization
- With classless addressing
  - the ISP can divide the prefix into several longer prefixes
- For instance, the ISP can divide a class C prefix into 4 longer prefixes
  - each one can accommodate a network of up to 62 hosts
    - all 0s and all 1s are reserved
- The original class C address has 8 bits of suffix
  - and each of the classless addresses has 6 bits of suffix
- Thus, instead of wasting addresses
  - ISP can assign each of the 4 classless prefixes to a subscriber

Classless Addressing: Example



Address Masks

- How can an IP address be divided at an arbitrary boundary?
- The classless and subnet addressing schemes require hosts and routers to store an additional piece of information:
  - a value that specifies the exact boundary between prefix and suffix
- To mark the boundary, IP uses a 32-bit value
  - known as an address mask, also called a subnet mask
- Why store the boundary size as a bit mask?
  - A mask makes processing efficient
- Hosts and routers need to compare the network prefix portion of the address to a value in their forwarding tables
  - The bit-mask representation makes the comparison efficient

Address Masks

- Suppose a router is given
  - a destination address, D
  - a network prefix represented as a 32-bit value, N
  - a 32-bit address mask, M
- Assume the top bits of N contain a network prefix, and the remaining bits have been set to zero
- To test whether the destination lies on the specified network, the router tests the condition:

  N == (D & M)
- The router
  - uses the mask with a "logical and (&)" operation to set the host bits of address D to zero (0)
  - and then compares the result with the network prefix N

Address Masks: Example

- Consider the following 32-bit network prefix:
  10000000 00001010 00000000 00000000 → 128.10.0.0
- Consider a 32-bit mask:
  11111111 11111111 00000000 00000000 → 255.255.0.0
- Consider a 32-bit destination address, which has a
  10000000 00001010 00000010 00000011 → 128.10.2.3
- A logical & between the destination address and the address mask extracts the high-order 16-bits
  10000000 00001010 00000000 00000000 → 128.10.0.0

CIDR Notation

- Classless Inter-Domain Routing (CIDR)
- Consider the mask needed for the example in Slide 23
  - It has 26 bits of 1s followed by 6 bits of 0s
  - In dotted decimal, the mask is: 255.255.255.192
- The general form of CIDR notation is: ddd.ddd.ddd.ddd/m
  - ddd is the decimal value for an octet of the address
  - m is the number of one bits in the mask
- Thus, one might write the following: 192.5.48.69/26
  - which specifies a mask of 26 bits

A CIDR Example

- Assume an ISP has the following block 128.211.0.0/16
- Suppose the ISP has 2 customers
  - one customer needs 12 IP addresses and the other needs 9
- The ISP can assign
  - customer1 CIDR: 128.211.0.16/28
  - customer2 CIDR: 128.211.0.32/28
  - both customers have the same mask size (28 bits), the prefixes differ

A CIDR Example (cont'd)

- The binary value assigned to customer1 is:
  - 10000000 11010011 00000000 00010000
- The binary value assigned to customer2 is:
  - 10000000 11010011 00000000 00100000
- There is no ambiguity
  - Each customer has a unique prefix
  - More important, the ISP retains most of the original address block
    - it can then allocate to other customers

CIDR Host Addresses

- Once an ISP assigns a customer a CIDR prefix
  - the customer can assign host addresses for its network users
- Suppose an organization is assigned 128.211.0.16/28
  - the organization will have 4-bits to use as a host address field
- Disadvantage of classless addressing
  - Because the host suffix can start on an arbitrary boundary, values are not easy to read in dotted decimal

CIDR Host Addresses

Network Prefix 128.211.0.16 / 28

0 … 28 … 31

1 0 0 0 0 0 0 0 : 1 1 0 1 0 0 1 1 : 0 0 0 0 0 0 0 0 : 0 0 0 1 | 0 0 0 0

Address Mask 255.255.255.240

0 … 28 … 31

1 1 1 1 1 1 1 1 : 1 1 1 1 1 1 1 1 : 1 1 1 1 1 1 1 1 : 1 1 1 1 | 0 0 0 0

Lowest Host Address 128.211.0.17

0 … 28 … 31

1 0 0 0 0 0 0 0 : 1 1 0 1 0 0 1 1 : 0 0 0 0 0 0 0 0 : 0 0 0 1 | 0 0 0 1

Highest Host Address 128.211.0.30

0 … 28 … 31

1 0 0 0 0 0 0 0 : 1 1 0 1 0 0 1 1 : 0 0 0 0 0 0 0 0 : 0 0 0 1 | 1 1 1 0

INDIRIZZI PRIVATI, SPECIALI E INDIRIZZI DEI ROUTER

Non tutti gli indirizzi IP sono utilizzabili, alcuni indirizzi hanno significato solo interno al computer e altri consentono di fare il bootstrap delle macchine prima che abbiano un indirizzo IP con cui comunicare.
I Router sono macchine con più indirizzi IP ... anche se non sempre con più interfacce fisiche di comunicazione.

Indirizzi pubblici e privati

- Non tutti gli indirizzi IP Unicast validi sono uguali
- Alcuni indirizzi sono stati definiti "privati" e non sono instradabili in Internet
  - Possono essere usati per costruire Intra-net private
- Un host con indirizzo IP privato ha bisogno di una apparato attivo che traduca opportunamente i suoi pacchetti per accedere a Internet
- NAT: Network Address Translator
  - Mappa la 5-tupla che identifica un flusso su un'altra 5-tupla con indirizzo pubblico, lavora a livello L3/L4
- Proxy
  - Gateway di L7, che interconnette a livello di singola applicazione

Indirizzi Privati

- 10.0.0.0 – 10.255.255.255
- 172.16.0.0 – 172.31.255.255
- 192.168.0.0 – 192.168.255.255

- Un indirizzo privato può essere riutilizzato in molti posti diversi
- All'interno di una stessa rete devono essere unici
- Normalmente sono usati tramite DHCP e non sono assegnati staticamente a un host

Special IP Addresses

- IP defines a set of special address forms that are reserved
  - That is, special addresses are never assigned to hosts
- Examples:
  - Network Address
  - Directed Broadcast Address
  - Limited Broadcast Address
  - This Computer Address
  - Loopback Address

Network Address

- It is convenient to have an address that can be used to denote the prefix assigned to a given network
- IP reserves host address zero
  - and uses it to denote a network
- Thus, the address 128.211.0.16/28 denotes a network
  - because the bits beyond the 28 are zero
  - 10000000 11010011 00000000 00010000
- A network address should never appear as the destination address in a packet

Directed Broadcast Address

- To simplify broadcasting (send to all)
  - IP defines a directed broadcast address for each physical network
- When a packet is sent to a network's directed broadcast
  - a single copy of the packet travels across the Internet
    - until it reaches the specified network
  - the packet is then delivered to all hosts on the network
- The directed broadcast address for a network is formed by adding a suffix that consists of all 1 bits to the network prefix
  - 10000000 11010011 00000000 00011111

Limited Broadcast Address

- Limited broadcast refers to a broadcast on a directly-connected network:
  - informally, we say that the broadcast is limited to a "single wire" meaning that it will never be forwarded by a router, even if the "wire" can be a huge Campus LAN
- Limited broadcast is used during system startup
  - by a computer that does not yet know the network number
- IP reserves the address consisting of 32-bits of 1s
  - 11111111 11111111 11111111 11111111
- Thus, IP will broadcast any packet sent to the all-1s address across the local network

This Computer Address

- A computer needs to know its IP address
  - before it can send or receive Internet packets
- TCP/IP contains protocols a computer can use to obtain its IP address automatically when the computer boots
  - … but the startup protocols also use an IP to communicate
- When using such startup protocols
  - a computer cannot supply a correct IP source address
  - To handle such cases IP reserves the address that consists of all 0s to mean this computer
  - 00000000 00000000 00000000 00000000

Loopback Address

- Loopback address used to test network applications
  - e.g., for preliminary debugging after a network application has been created
- A programmer must have two application programs that are intended to communicate across a network
- Instead of executing each program on a separate computer
  - the programmer runs both programs on a single computer
  - and instructs them to use a loopback address when communicating
- When one application sends data to another
  - data travels down the protocol stack to the IP software
  - then forwards it back up through the protocol stack to the second program

Loopback Address (cont'd)

- IP reserves the network prefix 127/8 for use with loopback
- The host address used with 127 is irrelevant
  - all host addresses are treated the same
  - programmers often use host number 1
  - so it makes 127.0.0.1 the most popular loopback address
- During loopback testing no packets ever leave a computer
  - the IP software forwards packets from one application to another
- The loopback address never appears in a packet traveling across a network

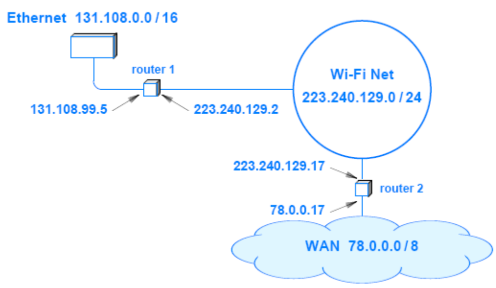
Summary of Special IP Addresses

Prefix	Suffix	Type Of Address	Purpose
all-0s	all-0s	this computer	used during bootstrap
network	all-0s	network	identifies a network
network	all-1s	directed broadcast	broadcast on specified net
all-1s	all-1s	limited broadcast	broadcast on local net
127 / 8	any	loopback	testing

Routers and IP Addresses

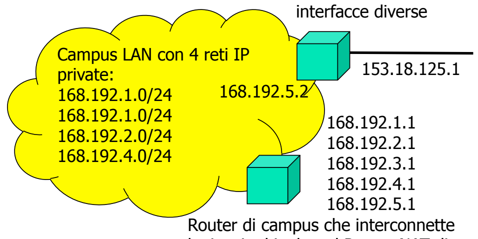
- Each router is assigned two or more IP addresses
  - one address for each logical network to which the router attaches
- To understand why, recall two facts:
  - A router connects multiple IP networks (by definition)
  - Each IP address contains a prefix that specifies a logical network
- A single IP address does not suffice for a router
  - because each router connects to multiple networks
  - and each network has a unique prefix
- The IP scheme can be explained by a principle:
  - An IP address does not identify a specific computer
  - each address identifies in interface, i.e., a logical connection between a computer and a network
  - A computer with multiple network connections (e.g., a router) must be assigned one IP address for each connection

An Example



Un altro esempio

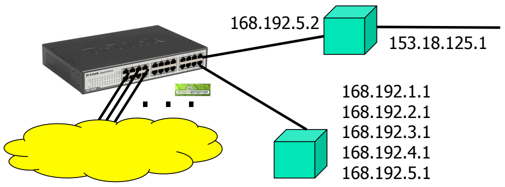
Router, NAT Firewall per accesso a Internet 2 indirizzi IP su due interfacce diverse



Router di campus che interconnette le 4 reti ed inoltra al Router NAT di collegamento verso Internet: 5 indirizzi IP tutti sulla stessa interfaccia fisica della campus LAN

Un altro esempio (cont)

Collegamento a livello ethernet



Router di campus che interconnette le 4 reti ed inoltra al Router NAT di collegamento verso Internet: 5 indirizzi IP tutti sulla stessa interfaccia fisica della campus LAN

ARP: ADDRESS RESOLUTION PROTOCOL

Protocollo di supporto a IP per mappare gli indirizzi IP sulle interfacce fisiche, ovvero sugli indirizzi MAC (Ethernet)

Address Resolution

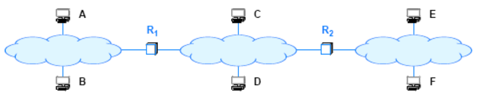
- A crucial step of the forwarding process requires a translation:
  - forwarding uses IP addresses
  - a frame transmitted must contain the MAC address of the next hop
  - IP must translate the next-hop IP address to a MAC address
- The principle is:
  - IP addresses are abstractions
    - provided by protocol software
  - Network does not know how to locate a computer from its IP address
    - the next-hop address must be translated to an equivalent MAC address

Address Resolution

- Translation from a computer's IP address to an equivalent hardware address is known as address resolution
  - And an IP address is said to be resolved to the correct MAC address
- Address resolution is local to a network
  - simple for Point-to-Point connections
  - need a protocol in the general case of shared access medium

Address Resolution

- One computer can resolve the address of another computer only if both computers attach to the same physical network
  - Direct delivery
  - A computer never resolves the address of a computer on a remote network
  - Address resolution is always restricted to a single network



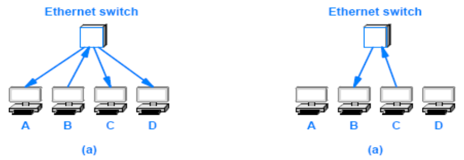
Address Resolution

- How can a host know if the address to resolve is local?
  - if it is local, the dest. IP address should have the same NetID (prefix) of the source IP address
- What happens if the address is not local?
  - Indirect delivery
  - Give the packet to a machine router should be on he way to the destination
  - ➔ topic of the next classes

Address Resolution Protocol (ARP)

- What algorithm does software use to translate?
  - The answer depends on the protocol and hardware addressing
    - here we are only concerned with the resolution of IP
- Most hardware has adopted the 48-bit Ethernet
- In Ethernet ➔ Address Resolution Protocol (ARP)

Address Resolution Protocol (ARP)



- Suppose B needs to resolve the IP address of C
- B broadcasts a request that says:
  - "I'm looking for the MAC address of a computer that has IP address C"
- The broadcast only travels across one network
- An ARP request message reaches all computers on a network
- When C receives a copy of the request it sends a directed reply back to B that says:
  - "I'm the computer with IP address C, and my MAC address is M"

ARP Message Format

- Rather than restricting ARP to IP and Ethernet
  - The standard describes a general form for ARP messages
  - It specifies how the format is adapted for each type of protocol
- Choosing a fixed size for a hardware address is not suitable
  - New network technologies might be invented that have addresses larger than the size chosen
  - The designers included a fixed-size field at the beginning of an ARP message to specify the size of the hardware addresses being used
- For example, when ARP is used with an Ethernet
  - the hardware address length is set to 6 octets
    - because an Ethernet address is 48 bits long

ARP Message Format

- To increase the generality of ARP
  - the designers also included an address length field
- ARP protocol can be used to bind an arbitrary high-level address to an arbitrary hardware address
- In practice, the generality of ARP is seldom used
  - most implementations of ARP are used to bind IP addresses to Ethernet addresses

ARP Message Format

0	8	16	24	31
HARDWARE ADDRESS TYPE		PROTOCOL ADDRESS TYPE		
HADDR LEN	PADDR LEN	OPERATION		
SENDER HADDR (first 4 octets)				
SENDER HADDR (last 2 octets)		SENDER PADDR (first 2 octets)		
SENDER PADDR (last 2 octets)		TARGET HADDR (first 2 octets)		
TARGET HADDR (last 4 octets)				
TARGET PADDR (all 4 octets)				

ARP Message Format

- HARDWARE ADDRESS TYPE
  - 16-bit field that specifies the type of hardware address being used
  - the value is 1 for Ethernet
- PROTOCOL ADDRESS TYPE
  - 16-bit field that specifies the type of protocol address being used
  - the value is 0x0800 for IPv4
- HADDR LEN
  - 8-bit integer that specifies the size of a hardware address in bytes
- PADDR LEN
  - 8-bit integer that specifies the size of a protocol address in bytes

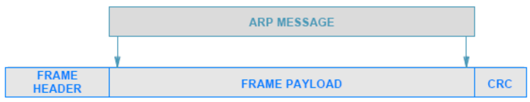
ARP Message Format

- OPERATION
  - 16-bit field that specifies whether the message
    - "request" (the field contains 1) or "response" (the field contains 2)
- SENDER HADDR
  - HADDR LEN bytes for the sender's hardware address
- SENDER PADDR
  - PADDR LEN bytes for the sender's protocol address
- TARGET HADDR
  - HADDR LEN bytes for the target's hardware address
- TARGET PADDR
  - PADDR LEN bytes for the target's protocol address

ARP Message Format

- An ARP message contains fields for two address bindings
  - one binding to the sender
  - other to the intended recipient, ARP calls it target
- When a request is sent
  - the sender does not know the target's hardware address (that is the information being requested)
    - field TARGET HADDR in an ARP request can be filled with zeroes
- In a response
  - the target binding refers to the initial computer that sent the request
  - Thus, the target address pair in a response serves no purpose
    - the inclusion of the target fields has survived from an early version of the protocol

ARP Encapsulation



- When it travels across a physical network an ARP message is encapsulated in a hardware frame
  - e.g., Ethernet
- An ARP message is treated as data being transported
  - the network does not parse the ARP message or interpret fields

ARP Encapsulation

- The type field in the frame header specifies that the frame contains an ARP message
- A sender must assign the appropriate value to the type field before transmitting the frame
- A receiver must examine the type field in each incoming frame
- Ethernet uses type field 0x806 to denote an ARP message
- The same value is used for both ARP requests/ responses
  - Frame type does not distinguish between types of ARP messages
  - A receiver must examine the OPERATION field in the message to determine whether an incoming message is a request or a response

ARP Caching and Message Processing

- Sending an ARP request for each datagram is inefficient
  - Three frames traverse the network for each datagram
    - an ARP request, ARP response, and the data datagram itself
- Most communications involve a sequence of packets
  - a sender is likely to repeat the exchange many times
- To reduce network traffic
  - ARP software extracts and saves the information from a response
    - so it can be used for subsequent packets
  - The software does not keep the information indefinitely
    - Instead, ARP maintains a small table of bindings in memory

ARP Caching and Message Processing

- ARP manages the table as a cache
  - an entry is replaced when a response arrives
  - the oldest entry is removed whenever the table runs out of space or after an entry has not been updated for a long period of time
  - ARP starts by searching the cache when it needs to bind an address

ARP Caching and Message Processing

- If the binding is present in the cache
  - ARP uses the binding without transmitting a request
- If the binding is not present in the cache
  - ARP broadcasts a request
  - waits for a response
  - updates the cache
  - and then proceeds to use the binding
- The cache is only updated when an ARP message arrives
  - either a request or a response
    - most computer communication involves two-way traffic; this can be exploited to same messages

ICMP: INTERNET CONTROL MESSAGE PROTOCOL

Messaggi di controllo, segnalazione, errore al livello IP

Internet Control Message Protocol

- IP includes a companion protocol, ICMP
  - It is used to report errors back to the original source
- IP and ICMP are co-dependent
  - IP depends on ICMP to report errors
  - and ICMP uses IP to carry error messages
- Many ICMP messages have been defined

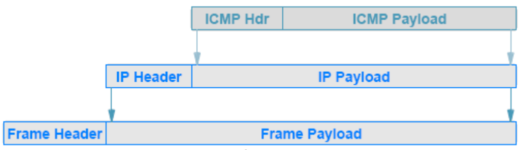
Internet Control Message Protocol

Number	Type	Purpose
0	Echo Reply	Used by the ping program
3	Dest. Unreachable	Datagram could not be delivered
5	Redirect	Host must change a route
8	Echo	Used by the ping program
11	Time Exceeded	TTL expired or fragments timed out
12	Parameter Problem	IP header is incorrect
30	Traceroute	Used by the traceroute program

Internet Control Message Protocol (ICMP)

- ICMP contains two message types:
  - messages used to report errors
    - e.g., Time Exceeded and Destination Unreachable
  - messages used to obtain information
    - e.g., Echo Request and Echo Reply
- Echo Request/Reply are used by the ping application to test connectivity
  - When a host receives an echo request message
    - ICMP software on a host or router sends an echo reply that carries the same data as the request

ICMP Message Format and Encapsulation



- ICMP uses IP to transport each error message:
  - when a router has an ICMP message to send
    - creates an IP datagram and encapsulates the ICMP message in it
  - the ICMP message is placed in the payload area of the IP datagram
  - the datagram is then forwarded as usual
    - with the complete datagram being encapsulated in a frame for transmission

ICMP handling

- ICMP messages do not have special priority
  - They are forwarded like any other datagram, with one minor exception
- If an ICMP error message causes an error
  - no error message is sent
- The reason should be clear:
  - the designers wanted to avoid the Internet becoming congested carrying error messages about error messages

DHCP: DYNAMIC HOST CONFIGURATION PROTOCOL

Come bootstrappare una rete senza dover configurare i singoli host

Protocol Parameters and Configuration

- Once a host or router has been powered on, OS is started and the protocol software is initialized
- How does the protocol software in a host or router begin operation?
- For a router, the configuration manager must specify initial values for items such as
  - the IP address for each network connection
  - the protocol software to run
  - and initial values for a forwarding table
  - the configuration is saved, and a router loads the values during startup
- Host configuration usually uses a two-step process, known as bootstrapping
  - A protocol was invented to allow a host to obtain multiple parameters with a single request, known as the Bootstrap Protocol (BOOTP)
  - Currently, DHCP is used to take care of most configuration needed

Dynamic Host Conf. Protocol (DHCP)

- Various mechanisms have been created to allow a host computer to obtain parameters
- An early mechanism known as the Reverse Address Resolution Protocol (RARP) allowed a computer to obtain an IP address from a server
- ICMP has Address Mask Request and Router Discovery messages
  - can obtain the address mask used and the address of a router
- Each of the early mechanisms was used independently
  - requests were broadcast and a host typically configured layers from lowest to highest
- DHCP allows a computer to join a new network and obtain an IP address automatically
  - The concept has been termed plug-and-play networking

Dynamic Host Conf. Protocol (DHCP)

- When a computer boots
  - the client computer broadcasts a DHCP Request
  - the server sends a DHCP Reply
    - DHCP uses the term offer to denote the message a server sends
    - and we say that the server is offering an address to the client
- We can configure a DHCP server to supply two types of addresses:
  - permanently assigned addresses as provided by BOOTP or
  - a pool of dynamic addresses to be allocated on demand
- Typically, a permanent address is assigned to a server, and a dynamic address is assigned to an arbitrary host
- In fact, addresses assigned on demand are not given out for an arbitrary length of time

Dynamic Host Conf Protocol (DHCP)

- DHCP issues a lease on the address for a finite period
  - The use of leases allows a DHCP server to reclaim addresses
- When the lease expires
  - the server places the address to the pool of available addresses
- When a lease expires, a host can choose to relinquish the address or renegotiate with DHCP to extend the lease
  - Negotiation occurs concurrent with other activity
- Normally, DHCP approves each lease extension
  - A computer continues to operate without any interruption
  - However, a server may be configured to deny lease extension for administrative or technical reasons
  - DHCP grants absolute control of leasing to a server
  - If a server denies an extension request
    - the host must stop using the address

DHCP Protocol Operation

- Recovery from loss or duplication
  - DHCP is designed to insure that missing or duplicate packets do not result in misconfiguration
  - If no response is received
    - a host retransmits its request
  - If a duplicate response arrives
    - a host ignores the extra copy
- Caching of a server address
  - once a host finds a DHCP server
    - the host caches the server's address
- Avoidance of synchronized flooding
  - DCHP takes steps to prevent synchronized requests

DHCP Message Format

- DHCP adopted a slightly modified version of the BOOTP message format
  - OP specifies whether the message is a Request or a Response
  - HTYPE and HLEN fields specify the network hardware type and the length of a hardware address
  - FLAGS specifies whether it can receive broadcast or directed replies
  - HOPS specifies how many servers forwarded the request
  - TRANSACTION IDENTIFIER provides a value that a client can use to determine if an incoming response matches its request
  - SECONDS ELAPSED specifies how many seconds have elapsed since the host began to boot
  - Except for OPTIONS (OP), each field in a DHCP message has a fixed size

DHCP Message Format

0 8 16 24 31

OP	HTYPE	HLEN	HOPS
TRANSACTION IDENTIFIER			
SECONDS ELAPSED		FLAGS	
CLIENT IP ADDRESS			
YOUR IP ADDRESS			
SERVER IP ADDRESS			
ROUTER IP ADDRESS			
CLIENT HARDWARE ADDRESS (16 OCTETS) ⋮			
SERVER HOST NAME (64 OCTETS) ⋮			
BOOT FILE NAME (128 OCTETS) ⋮			
OPTIONS (VARIABLE) ⋮			

DHCP Message Format

- Later fields in the message are used in a response to carry information back to the host that sent a request
  - if a host does not know its IP address, the server uses field YOUR IP ADDRESS to supply the value
  - server uses fields SERVER IP ADDRESS and SERVER HOST NAME to give the host information about the location of a server
  - ROUTER IP ADDRESS contains the IP address of a default router
- DHCP allows a computer to negotiate to find a boot image
  - To do so, the host fills in field BOOT FILE NAME with a request
  - The DHCP server does not send an image

ROUTING: PRINCIPI E SCOPI

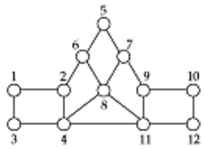
Come si deriva e calcola la topologia di Internet?
Data la topologia, come trovo il cammino migliore tra una sorgente e una destinazione?

Routing: What is it?

- Process of finding a path from a source to every destination in the network
- Suppose you want to connect to Antarctica from your desktop
  - what route should you take?
  - does a shorter route exist?
  - what if a link along the route goes down?
  - what if you're on a mobile wireless link?
- Routing deals with these types of issues

Basics

- A routing protocol sets up a routing table in routers
  - internal table that says, for each destination, which is the next output to take
- A node makes a local choice depending on global topology: this is the fundamental problem



ROUTING TABLE AT 1

Destination	Next hop	Destination	Next hop
1	—	7	2
2	2□	8□	2□
3	3□	9□	2□
4	3□	10□	2□
5	2□	11□	3□
6	2	12	3

Key problem

- How to make correct local decisions?
  - each router must know something about global state
- Global state
  - inherently large
  - dynamic
  - hard to collect
- A routing protocol must intelligently summarize relevant information

Requirements

- Minimize routing table space
  - fast to look up
  - less to exchange
- Minimize number and frequency of control messages
- Robustness: avoid
  - black holes
  - loops
  - oscillations
- Use optimal path

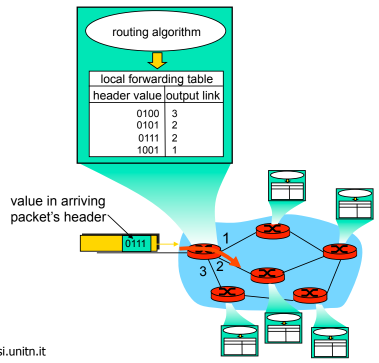
Different degrees of freedom

- Centralized vs. distributed routing
  - centralized is simpler, but prone to failure and congestion
- Global vs local information exchange
  - convey global information is expensive
- Static vs dynamic
  - static may work at the edge, not in the core
- Stochastic vs. deterministic
  - stochastic spreads load, avoiding oscillations, but misorders
- Single vs. multiple path
  - primary and alternative paths (compare with stochastic)
- State-dependent vs. state-independent
  - do routes depend on current network state (e.g. delay)

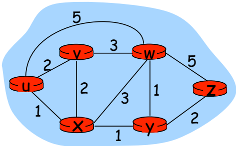
Dynamic Routing And Routers

- To ensure that all routers maintain information about how to reach each possible destination
  - each router uses a route propagation protocol
    - to exchange information with other routers
  - when it learns about changes in routes
    - updates the local routing table
- Because routers exchange information periodically
  - the local routing table is updated continuously

Interplay between routing, forwarding



Graph abstraction

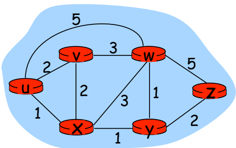


Graph: G = (N,E)

N = set of routers = { u, v, w, x, y, z }

E = set of links ={ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

Graph abstraction: costs



- c(x,x') = cost of link (x,x')
  - e.g., c(w,z) = 5
- cost could always be 1, or inversely related to bandwidth, or inversely related to congestion

Cost of path $(x_1, x_2, x_3,\ldots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

Question: What's the least-cost path between u and z ?

Routing algorithm: algorithm that finds least-cost path

Distance Vector Algorithms

Consistency criterion

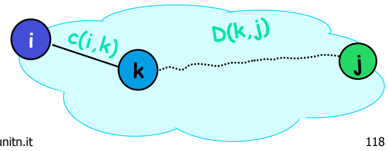
Define

c(i,k) := cost from i to k (direct connection)

D(i,j) := cost of least-cost path from i to j

➔ The subset of a shortest path is also the shortest path between the two intermediate nodes

- Then, if the shortest path from node i to node j, with distance D(i,j), passes through neighbor k, with link cost c(i,k), we have:
  - D(i,j) = c(i,k) + D(k,j)



Distance Vector (DV) algorithm

- Initial distance values (iteration 1):
  - D(i,i) = 0 ;
  - D(i,k) = c(i,k) if k is a neighbor (i.e. k is one-hop away); and
  - D(i,j) = INFINITY for all other non-neighbors j.
- Note that the set of values D(i,*) is a distance vector at node i.
- The algorithm also maintains a next-hop value (forwarding table) for every destination j, initialized as:
  - next-hop(i) = i;
  - next-hop(k) = k if k is a neighbor, and
  - next-hop(j) = UNKNOWN if j is a non-neighbor.

Distance Vector (DV) algorithm

- After every iteration each node i exchanges its distance vectors D(i,*) with its immediate neighbors.
- For any neighbor k, if c(i,k) + D(k,j) < D(i,j), then:
  - D(i,j) = c(i,k) + D(k,j)
  - next-hop(j) = k

In summary

Basic idea:

- From time-to-time, each node sends its own distance vector estimate to neighbors

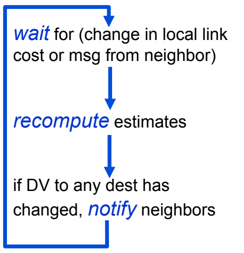
Asynchronous

- When a node x receives new DV estimate from neighbor, it updates its own DV using B-F equation:
  $D(x,y) \leftarrow \min_v\{c(x,v) + D(v,y)\}$ for each node $y \in N$
- Under minor, natural conditions, the estimate D(x,y) converges to the actual least cost

In summary

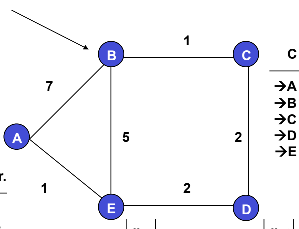
- Iterative, asynchronous: each local iteration caused by:
  - local link cost change
  - DV update message from neighbor
- Distributed: each node notifies neighbors only when its DV changes
  - neighbors then notify their neighbors if necessary

Each node:



Distance Vector: example (starting point)

B	dist	thr.
→A	7	A
→B	0	-
→C	1	C
→D	-	-
→E	5	E



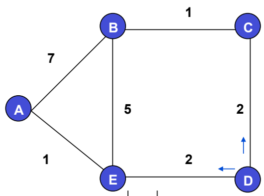
C	dist	thr.
→A	-	-
→B	1	B
→C	0	-
→D	2	D
→E	-	-

A	dist	thr.
→A	0	-
→B	7	B
→C	-	-
→D	-	-
→E	1	E

E	dist	thr.
→A	1	A
→B	5	B
→C	-	-
→D	2	D
→E	0	-

D	dist	thr.
→A	-	-
→B	-	-
→C	2	C
→D	0	-
→E	2	E

Distance Vector: example (running)

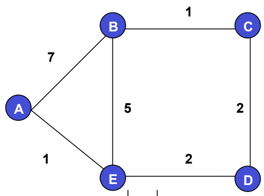


C	dist	thr.
→A	-	-
→B	1	B
→C	0	-
→D	2	D
→E	-	-

E	dist	thr.
→A	1	A
→B	5	B
→C	-	-
→D	2	D
→E	0	-

D	dist	thr.
→A	-	-
→B	-	-
→C	2	C
→D	0	-
→E	2	E

Distance Vector: example (running)



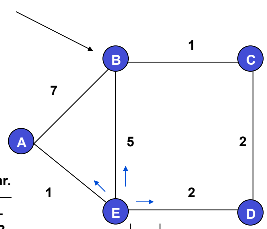
C	dist	thr.
→A	-	-
→B	1	B
→C	0	-
→D	2	D
→E	4	D

E	dist	thr.
→A	1	A
→B	5	B
→C	4	D
→D	2	D
→E	0	-

D	dist	thr.
→A	-	-
→B	-	-
→C	2	C
→D	0	-
→E	2	E

Distance Vector: example (running)

B	dist	thr.
→A	7	A
→B	0	-
→C	1	C
→D	-	-
→E	5	E



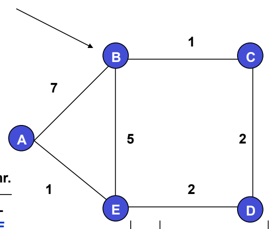
A	dist	thr.
→A	0	-
→B	7	B
→C	-	-
→D	-	-
→E	1	E

E	dist	thr.
→A	1	A
→B	5	B
→C	4	D
→D	2	D
→E	0	-

D	dist	thr.
→A	-	-
→B	-	-
→C	2	C
→D	0	-
→E	2	E

Distance Vector: example (running)

B	dist	thr.
→A	6	E
→B	0	-
→C	1	C
→D	7	E
→E	5	E



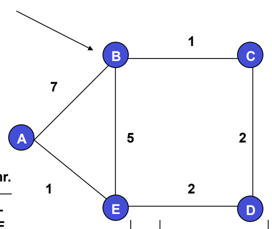
A	dist	thr.
→A	0	-
→B	6	E
→C	5	E
→D	3	E
→E	1	E

E	dist	thr.
→A	1	A
→B	5	B
→C	4	D
→D	2	D
→E	0	-

D	dist	thr.
→A	3	E
→B	7	E
→C	2	C
→D	0	-
→E	2	E

Distance Vector: example (final point)

B	dist	thr.
→A	6	E
→B	0	-
→C	1	C
→D	3	C
→E	5	E



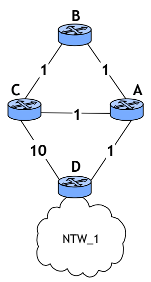
C	dist	thr.
→A	5	D
→B	1	B
→C	0	-
→D	2	D
→E	4	D

A	dist	thr.
→A	0	-
→B	6	E
→C	5	E
→D	3	E
→E	1	E

E	dist	thr.
→A	1	A
→B	5	B
→C	4	D
→D	2	D
→E	0	-

D	dist	thr.
→A	3	E
→B	3	C
→C	2	C
→D	0	-
→E	2	E

Problem: "counting to infinity"



Router A

Dest	Next	Metric
NTW_1	D	2

Router B

Dest	Next	Metric
NTW_1	A	3

Router C

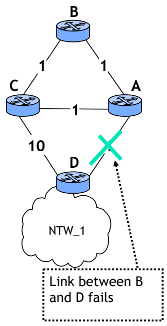
Dest	Next	Metric
NTW_1	A	3

Router D

Dest	Next	Metric
NTW_1	dir	1

- Consider the entries in each routing table for network NTW_1
- Router D is directly connected to NTW_1

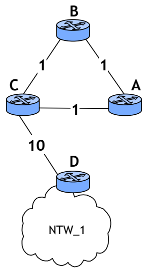
Problem: "counting to infinity"



time →

Router	Dest	Next	Metric	Dest	Next	Metric	Dest	Next	Metric
Router A	NTW_1	Unr.	-	NTW_1	C	4	NTW_1	C	5
Router B	NTW_1	A	3	NTW_1	C	4	NTW_1	C	5
Router C	NTW_1	A	3	NTW_1	B	4	NTW_1	B	5
Router D	NTW_1	dir	1	NTW_1	dir	1	NTW_1	dir	1

Problem: "counting to infinity"



time →

...

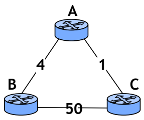
Router	Dest	Next	Metric	Dest	Next	Metric
Router A	NTW_1	C	11	NTW_1	C	12
Router B	NTW_1	C	11	NTW_1	C	12
Router C	NTW_1	B	11	NTW_1	D	11
Router D	NTW_1	dir	1	NTW_1	dir	1

Solution to "counting to infinity"

- Maximum number of hops bounded to 15
  - this limits the convergence time
- Split Horizon
  - simple
    - each node *omits* routes learned from one neighbor in update sent to that neighbor
  - with poisoned reverse
    - each node *include* routes learned from one neighbor in update sent to that neighbor, setting their metrics to infinity
      - drawback: routing message size greater than simple Split Horizon

Distance Vector: link cost changes

- If link cost changes:
  - good news travels fast
    - good = cost decreases
  - bad news travels slow
    - bad = cost increases



- Exercise
  - try to apply the algorithm in the simple scenario depicted above when
    - the cost of the link A → B changes from 4 to 1
    - the cost of the link A → B changes from 4 to 60

RIP at a glance

- **R**outing **I**nformation **P**rotocol
- A simple intradomain protocol
- Straightforward implementation of Distance Vector Routing…
  - Distributed version of Bellman-Ford (DBF)

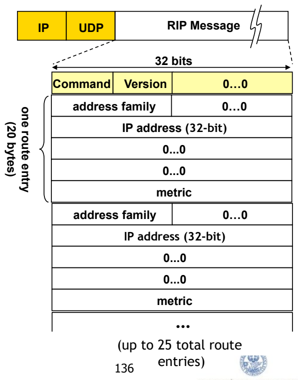
  …with well known issues
  - slow convergence
  - works with limited network size
- Strengths
  - simple to implement
  - simple management
  - widespread use

RIP at a glance

- Metric based on hop count
  - maximum hop count is 15, with "16" equal to "∞"
    - imposed to limit the convergence time
  - the network administrator can also assign values higher than 1 to a single hop
- Each router advertises its distance vector every 30 seconds (or whenever its routing table changes) to all of its neighbors
  - RIP uses UDP, port 520, for sending messages
- Changes are propagated across network
- Routes are timeout (set to 16) after 3 minutes if they are not updated

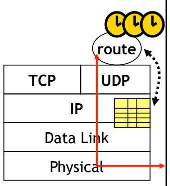
RIP: Message Format

- Command: 1=request 2=response
  - Updates are replies whether asked for or not
  - Initializing node broadcasts request
  - Requests are replied to immediately
- Version: 1
- Address family: 2 for IP
- IP address: non-zero network portion, zero host portion
  - Identifies particular network
- Metric
  - Path distance from this router to network
  - Typically 1, so metric is hop count



RIP procedures: introduction

- RIP routing tables are managed by application-level process
  - e.g., *routed* on UNIX machines
- Advertisements are sent in UDP packets (port 520)
- RIP maintains 3 different timers to support its operations
  - Periodic update timer (25-30 sec)
    - used to sent out update messages
  - Invalid timer (180 sec)
    - If update for a particular entry is not received for 180 sec, route is invalidated
  - Garbage collection timer (120 sec)
    - An invalid route in marked, not immediately deleted
    - For next 120 s. the router advertises this route with distance infinity

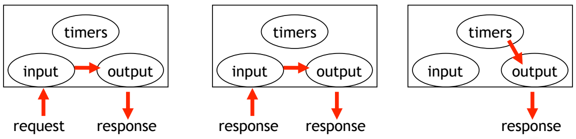


RIP procedures: input processing

- Request Messages
  - they may arrive from routers which have just come up
  - action: the router responds directly to the requestor's address and port
    - request is processed entry by entry
- Response Messages
  - they may arrive from routers that perform regular updates, triggered updates or respond to a specific query
  - action: the router updates its routing table
    - in case of new route or changed routes, the router starts a triggered update procedure

RIP procedures: output processing

- Output are generated
  - when the router comes up in the network
  - if required by the input processing procedures
  - by regular routing update
- Action: the router generates the messages according to the commands received
  - the messages contain entries from the routing table

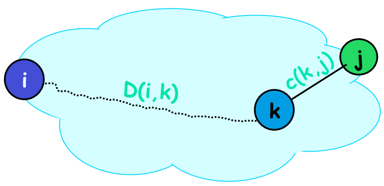


ALGORITMI LINK-STATE

Un approccio diverso e "centralizzato" per trovare i cammini minimi

Link State (LS) Approach

- The link state (Dijkstra) approach is iterative, but it pivots around destinations j, and their predecessors k = p(j)
  - Observe that an alternative version of the consistency condition holds for this case: D(i,j) = D(i,k) + c(k,j)



- Each node i collects all link states c(*,*) first and runs the complete Dijkstra algorithm locally.

Link State (LS) Approach…

- After each iteration, the algorithm finds a new destination node j and a shortest path to it.
- After m iterations the algorithm has explored paths, which are m hops or smaller from node i.
  - It has an m-hop view of the network just like the distance-vector approach
- The Dijkstra algorithm at node i maintains two sets:
  - set N that contains nodes to which the shortest paths have been found so far, and
  - set M that contains all other nodes.
  - For all nodes k, two values are maintained:
    - D(i,k): current value of distance from i to k.
    - p(k): the predecessor node to k on the shortest known path from i

Dijkstra: Initialization

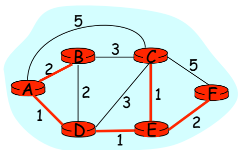
- Initialization:
  - D(i,i) = 0 and p(i) = i;
  - D(i,k) = c(i,k) and p(k) = i if k is a neighbor of I
  - D(i,k) = INFINITY and p(k) = UNKNOWN if k is not a neighbor of I
  - Set N = { i }, and next-hop (i) = I
  - Set M = { j | j is not i}
- Initially set N has only the node i and set M has the rest of the nodes.
- At the end of the algorithm, the set N contains all the nodes, and set M is empty

Dijkstra: Iteration

- In each iteration, a new node j is moved from set M into the set N.
  - Node j has the minimum distance among all current nodes in M, i.e. D(i,j) = min {l ε M} D(i,l).
  - If multiple nodes have the same minimum distance, any one of them is chosen as j.
  - Next-hop(j) = the neighbor of i on the shortest path
    - Next-hop(j) = next-hop(p(j)) if p(j) is not i
    - Next-hop(j) = j if p(j) = i
  - Now, in addition, the distance values of any neighbor k of j in set M is reset as:
    - If D(i,k) < D(i,j) + c(j,k), then
      D(i,k) = D(i,j) + c(j,k), and p(k) = j.
- This operation is called "relaxing" the edges of node j.

Dijkstra's algorithm: *example*

Step	set N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
0	A	2,A	5,A	1,A	infinity	infinity
1	AD	2,A	4,D		2,D	infinity
2	ADE	2,A	3,E			4,E
3	ADEB		3,E			4,E
4	ADEBC					4,E
5	ADEBCF					



The shortest-paths spanning tree rooted at A is called an SPF-tree

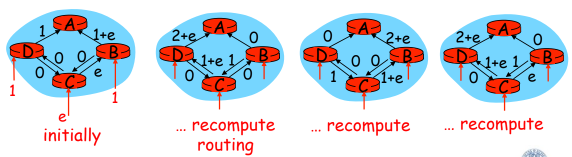
Dijkstra's algorithm, discussion

Algorithm complexity: n nodes

- each iteration: need to check all nodes, w, not in N
- n(n+1)/2 comparisons: $O(n^2)$
- more efficient implementations possible: O(n log(n))

Oscillations possible:

- e.g., link cost = amount of carried traffic



Summary: Distributed Routing Techniques

Link State

- Topology information is flooded within the routing domain
- Best end-to-end paths are computed locally at each router.
- Best end-to-end paths determine next-hops.
- Based on minimizing some notion of distance
- Works only if policy is shared and uniform
- Examples: OSPF

Distance Vector

- Each router knows little about network topology
- Only best next-hops are chosen by each router for each destination network.
- Best end-to-end paths result from composition of all next-hop choices
- Does not require any notion of distance
- Does not require uniform policies at all routers
- Examples: RIP

Comparison of LS and DV algorithms

Message complexity

- LS: with n nodes, E links, O(nE) msgs sent
- DV: exchange between neighbors only
  - convergence time varies

Speed of Convergence

- LS: O(n2) algorithm requires O(nE) msgs
  - may have oscillations
- DV: convergence time varies
  - may be routing loops
  - count-to-infinity problem

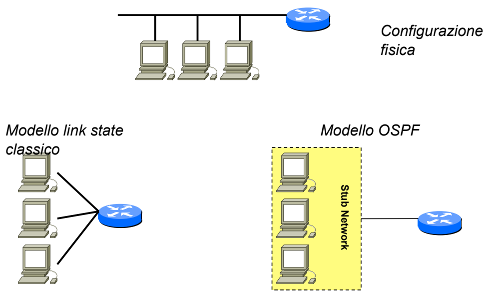
Robustness: what happens if router malfunctions?

- 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
    - error propagate thru network

Open Shortest Path First (OSPF)

- In alternativa al protocollo RIP di tipo Distance Vector in Internet esiste il protocollo OSPF di tipo Link State
- I tre principali criteri di progettazione del protocollo OSPF sono:
  - distinzione tra host e router
  - reti broadcast
  - suddivisione delle reti di grandi dimensioni
- Hli host sono collocati nelle aree periferiche della rete a sottoreti locali connesse alla attraverso router (default gateway)
- Il modello link state prevede che il database *link state* includa una entry per ogni link tra host e router
- OSPF associa il link di accesso ad una *stub network*
  - una stub network è una sottorete terminale che non fornisce servizio di transito
  - il link di accesso viene identificato dall'indirizzo della sottorete

Distinzione host/router (2)



Il protocollo OSPF

- Il protocollo OSPF utilizza 3 procedure, chiamati ancora `protocolli', per svolgere le proprie funzioni
  - Hello Protocol
  - Exchange Protocol
  - Flooding Protocol

Messaggi OSPF (1)

- I messaggi OSPF sono trasportati direttamente all'interno dei pacchetti IP
  - non viene utilizzato il livello di trasporto
  - nelle reti broadcast biene usato un indirizzo multicast
- Tutti i messaggi OSPF condividono lo stesso header

Version #	*Type*	*Packet length*	
Router ID			
Area ID			
Checksum		*Auth Type*	
Authentication			
Authentication			

Messaggi OSPF (2)

- Version # = 2
- Type: indica il tipo di messaggio
- Packet Length: numero di byte del messaggio
- Router ID: indirizzo IP del router di riferimento

Version #	*Type*	*Packet length*	
Router ID			
Area ID			
Checksum		*Auth Type*	
Authentication			
Authentication			

Messaggi OSPF (3)

- Area ID: identificativo dell'area
  - 0 per la Bacvbone area
- Auth Type: tipo di autenticazione
  - 0 no autenticazione, 1 autenticazione con passwd
- Authentication: password

Version #	*Type*	*Packet length*
Router ID		
Area ID		
Checksum		*Auth Type*
Authentication		
Authentication		

Il protocollo Hello

- Funzioni:
  - verificare l'operatività dei link
- Messaggi:
  - Hello

Common header (type = 1, hello)		
Network mask		
Hello interval	*Options*	*Priority*
Dead interval		
Designated router		
Backup Designated router		
Neighbor		

Hello Protocol: formato pacchetto (3)

- Neighbor: lista di nodi adiacenti da cui ha ricevuto un messaggio di Hello negli ultimi **dead interval** secondi

Common header (type = 1, hello)		
Network mask		
Hello interval	*Options*	*Priority*
Dead interval		
Designated router		
Backup Designated router		
Neighbor		

Il protocollo Exchange

- Funzioni:
  - sincronizzazione dei database link state (bring up adjacencies) tra due router che hanno appena verificato l'operatività bidirezionale del link che li connette
  - protocollo client-server
  - messaggi:
    - Database Description Packets
    - Link State Request
    - Link State Update
  - N.B. il messaggio Link State Update viene distribuito in flooding

Exchange Protocol: messaggi (1)

- Database Description

Common header (type = 2, db description)			
0	*0*	*Options*	*0*
DD sequence number			
Link State Type			
Link State ID			
Advertising router			
Link State Sequence Number			
Link State Checksum		*Link State Age*	

Exchange Protocol: messaggi (2)

- Link State Request

Common header (type = 3, link state request)
Link State Type
Link State ID
Advertising router

- Link state Update

Common header (type = 4, link state update)
Number of link state advertisement
Link state advertisement #1
Link state advertisement #2

Il protocollo di Flooding

- Funzioni:
  - aggiornare il database link state dell'autonomous system a seguito del cambiamento di stato di un link
  - Garantisce la consegna di tutti I messaggi a tutti, a costo di parecchie repliche
- Messaggi:
  - Link State Update

Common header (type = 4, link state update)
Number of link state advertisement
Link state advertisement #1
Link state advertisement #2