INTERNET TCP/IP

Modello OSI

7	**Applicazione**
6	**Presentazione**
5	**Sessione**
4	**Trasporto**
3	**Rete**
2	**Data Link**
1	**Fisico**

Livello 7 **Applicazione:** esempi di applicazioni sono la posta elettronica e il trasferimento dei file

Livello 6 **Presentazione:** rappresentazione, compressione e crittografia dei dati

Livello 5 **Sessione:** la chiamata di procedura remota è un esempio particolare di sessione

Livello 4 **Trasporto:** comunicazione "*end-to-end*", virtualizzazione del collegamento di rete fra trasmittente e ricevente

Livello 3 **Rete:** instradamento dei frame, interconnessione di reti locali e geografiche, gestione delle situazioni di congestione

Livello 2 **Collegamento dati** ("*frame*"): riconoscimento e ritrasmissione di frame affetti da errori, controllo di flusso)

Livello 1 **Fisico** (mezzo trasmissivo): modalità di codifica dei dati e di sincronizzazione a basso livello della sincronizzazione

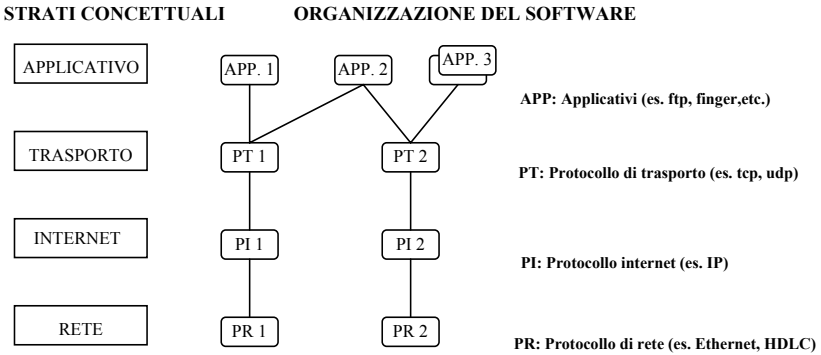
Dal Modello OSI a 7 livelli

Modello OSI semplificato

4	**Processo**
3	**Trasporto**
2	**Rete**
1	**Data Link**

Livelli della suite TCP/IP

4	**Livello applicativo**
3	**Livello di trasporto**
2	**Livello internet**
1	**Interfaccia di rete**



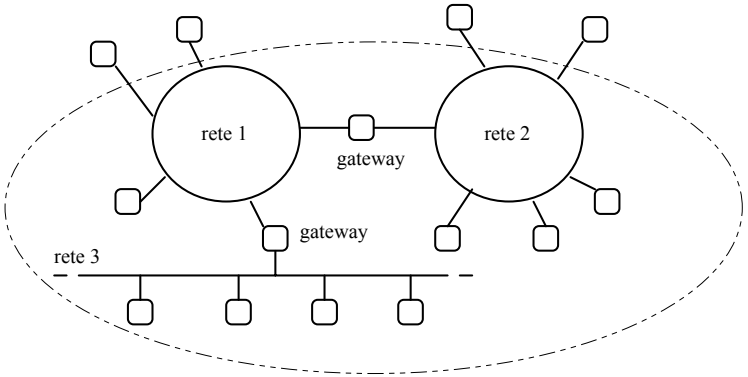
INTERNETWORKING

TCP/IP **T**ransport **C**ontrol **P**rotocol/**I**nternet **P**rotocol
DARPA con supporto del DoD
Defense Advanced Projects Research Agency
SNA **S**ystem **N**etwork **A**rchitecture IBM
DNA **D**igital **N**etwork **A**rchitecture DEC
XNS **X**erox **N**etwork **S**ystem

Nel caso delle reti che ci interessano *tipicamente*
Livelli fino al DATA LINK
Uso di qualunque protocollo standard o meno

ETHERNET (CSMA/CD bus)
standard di connessione con **unico bus a 10Mbps**
- connettore a *basso costo:* cavo coassiale
  cavo coassiale (transceiver) o doppino (in hub)
- invio/ricezione packet-switching di *messaggi*
- supporto diretto *broadcast / multicast*

Interconnessione di reti



Collegamento di reti diverse attraverso gateway
gateway connette diversi tipi di rete effettuando le necessarie conversioni di protocollo

RIUSO

non si progetta una rete nuova;
si sfruttano **reti esistenti** (flessibilità, abbattimento dei costi, tempi brevi di installazione)
i pacchetti nel percorso dal sorgente al destinatario, attraversano reti intermedie con collegamenti replicati
gli utenti non devono nè essere influenzati, nè venire a conoscenza di un traffico extra sulle loro reti locali
Principio di MINIMA INTRUSIONE

Trasparenza e Dinamicità

INTERNETWORKING

***Problema*:**
necessità di una interconnessione universale
requisiti di **eterogeneità**
impossibilità di servire tutti gli utenti con una singola rete (esigenze contrastanti: distanza, velocità)

Soluzione:
Interconnessione di reti, cooperante, unificata per realizzare un servizio di **comunicazione universale**

***Requisiti*:**
accomodare tutte le nuove tecnologie (per connessione di reti tecnologicamente diverse)
progettare nuovo software di comunicazione indipendente dalla tecnologia e dai programmi applicativi per rete virtuale

NOTA: ipotesi di servizio al meglio (best-effort)

Internet

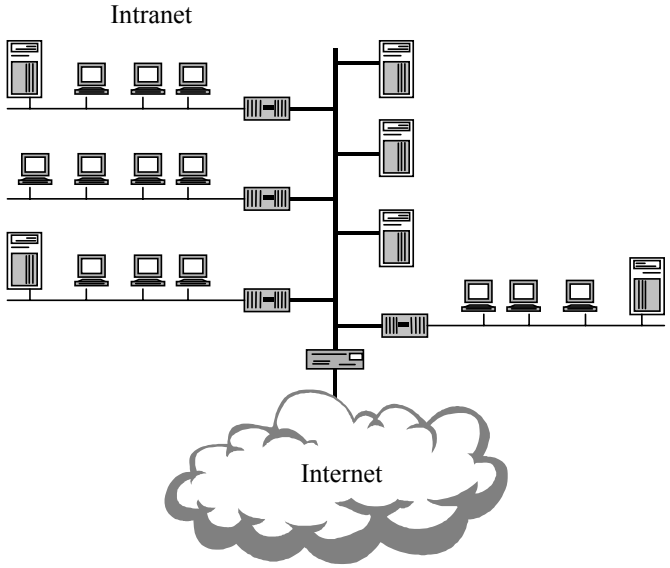
Tutte le reti interconesse in una unica globalità
(***il migliore dei mondi possibili***)

SISTEMA GLOBALE

Intranet

Una rete o un insieme di reti interconnesse (località) con esigenze di

ottimizzazione per le operazioni **locali**
comunicazione con il sistema **globale**



come sono collegate le tratte?

Chi paga per i servizi?
servizi da pagare con qualità
accounting, billing, ...
in base a conoscenza dell'utente

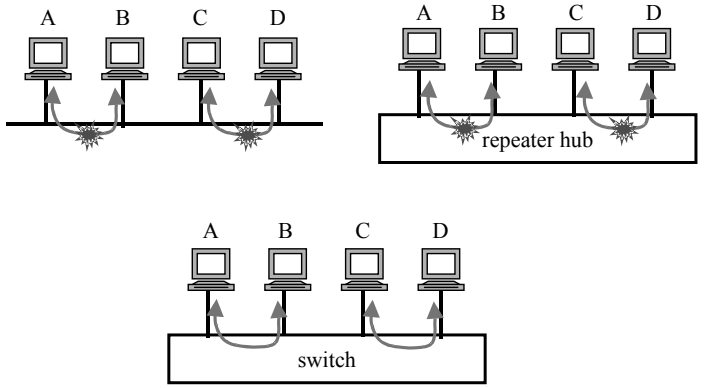
Terminologia in interconnessione

ripetitori **rigenerano un segnale a livello fisico**
oltre un definito livello di attenuazione ==> ripetitore

Problemi di **carico del sistema**
il ripetitore non effettua separazione delle tratte

gli **hub** forniscono una soluzione tipo ripetitori a basso costo

Gli **switch** sono in grado di gestire connessioni dinamiche su necessità



Gli **switch** evolvono nel senso della intelligenza
costo
performance

BRIDGE

bridge **collegano una rete ad un'altra** con capacità di **separazione** e maggiore **intelligenza**

livello di data link

due reti omogenee sono controllate da un **bridge** che bufferizza e passa i frame dall'una all'altra, solo se necessario, e al controllo di errore

separazione effettiva delle reti
bufferizzazione dei frame (caso di overflow)
capacità di gestire controlli di accesso diversi
monitoring della rete
performance ed affidabilità

ritardo di bufferizzazione
bufferizzazione non infinita
trasformazione dei frame (con controllo)

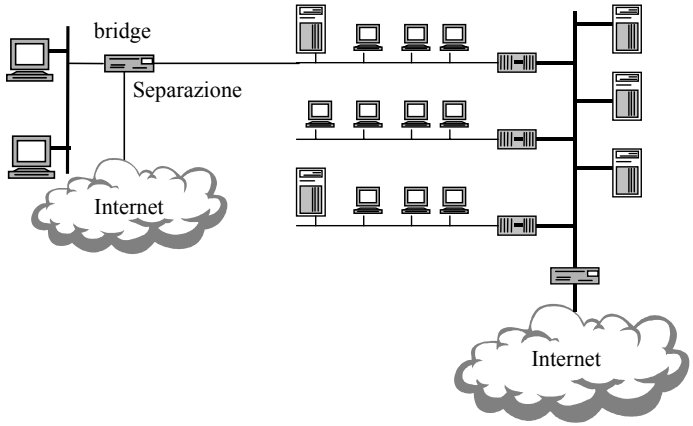
bridge multiporta

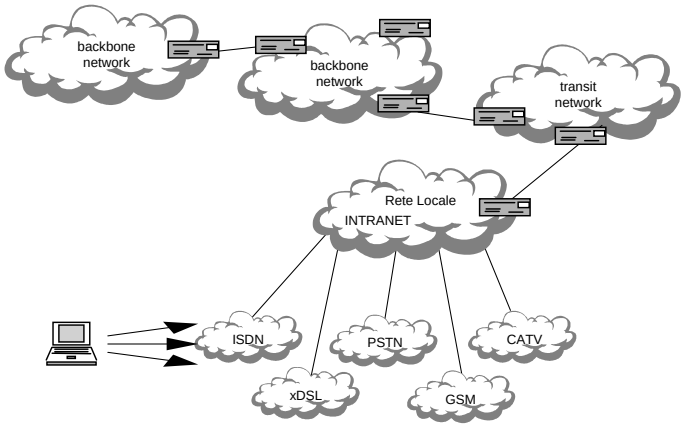
con più segmenti di rete connessi

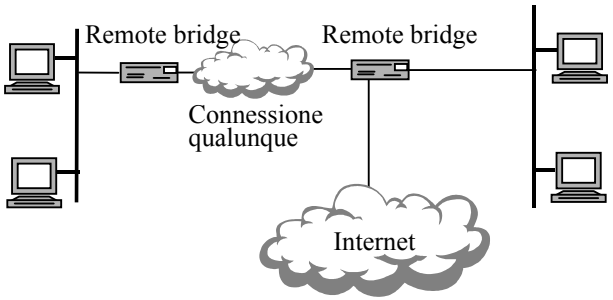
Un bridge connette almeno due reti, ma può anche connetterne diverse

stabilendo connessioni e
riservando bande
su richiesta o meno

Reti Interconnesse







Bridge trasparenti

sono invisibili all'utilizzatore

si realizza un routing isolato

con database di forwarding
o in PROM
o con capacità di **apprendimento**

Interconnessione usate per l'intero sistema

Il bridge impara la allocazione delle stazioni vedendo il traffico della rete e dai vicini

la fase di orientamento avviene inizialmente o per ogni variazione

FASE di LEARNING iniziale

Alla **inizializzazione** (inserimento)

un bridge comincia a vedere che sta facendo routing nel sistema e si adegua

Possibilità di conflitti

Algoritmo **spanning tree**

per determinare una gerarchia che impedisce conflitti

i **bridge** devono costruire un **albero** sul *grafo globale di interconnessioni*

scambiando messaggi per trovare i costi più bassi di collegamento e determinare i ruoli dei bridge

Si sceglie un **bridge radice** e ognuno trova il cammino minimo (passi e velocità)

La connessione ideale creata tra i **bridge** è l'albero che percorre tutto la topologia (con una sola radice)

I Bridge si scambiano informazioni (Bridge Protocol Data Unit) secondo le loro esigenze

source routing bridge

Il routing viene fatto in modo non trasparente
con costo elevato ma flessibilità (vedi IP source routing)

bridge remoti

collegamento dedicato tra punti geograficamente lontani

attraverso reti pubbliche packet-switching o linee dedicate

Backbone

collegamento veloce tra sottoreti diverse

uso di interconnessioni ad alta velocità (FDDI)

router (o **gateway**)

sistema per il passaggio da una rete ad un'altra con obiettivo di routing (livello network)

protocol converter

sistemi che collegano reti diverse a più alto livello con protocolli diversi di interconnessione

Il problema della **separazione** tra reti è
diventato dominante vista la crescita esponenziale
delle reti interconnesse

Router vs. bridge

maggiore separazione e decisioni diverse per cammini
gestione database separati per le reti
identificazione e gestione errori

STACK TCP

livello di TRASPORTO

TCP Transmission Control Protocol

flusso di byte bidirezionale a
canale virtuale best effort, **dati**
non duplicati, **affidabili**, con **controllo di flusso**

UDP User Datagram Protocol

Scambio di **messaggi**

livello di RETE

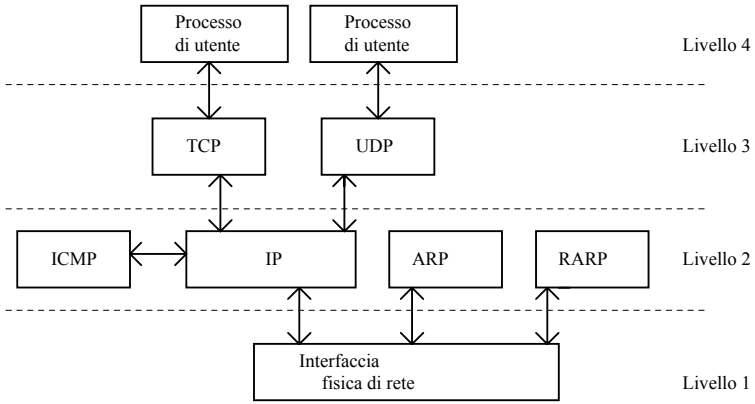
IP Internet Protocol

Scambio di datagrammi senza garanzia di consegna

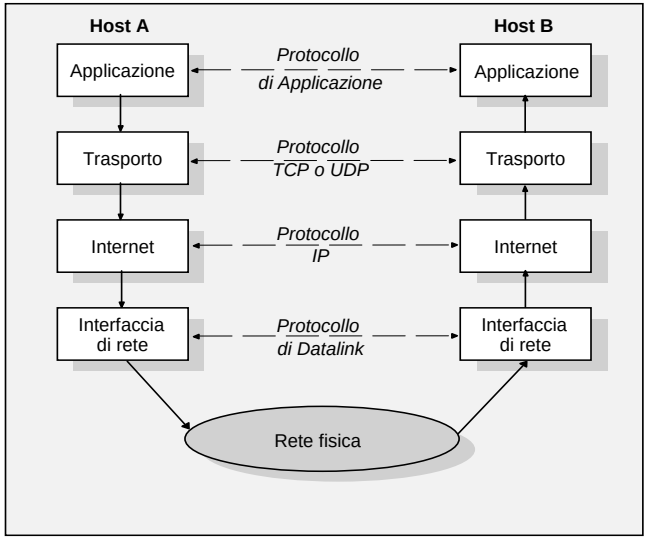
gestione di RETE

ICMP Internet Control Message Protocol

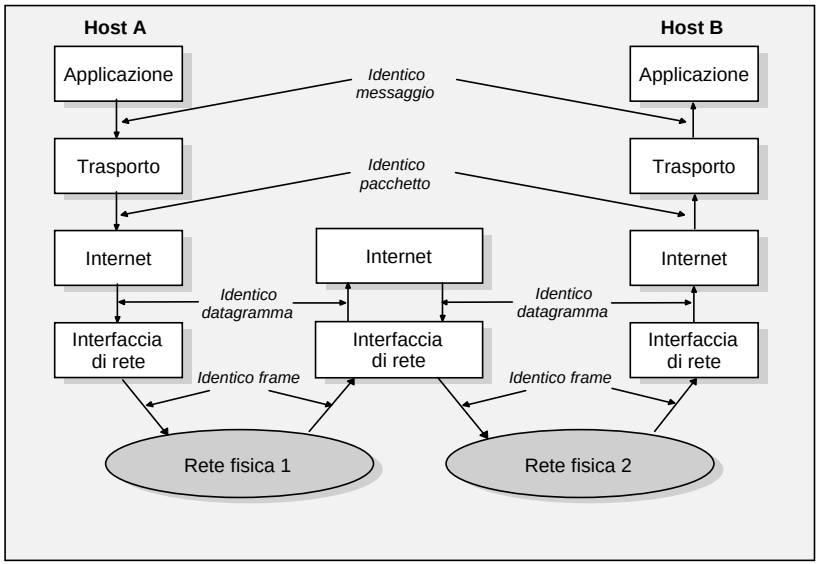
Scambio messaggi di controllo



Applicazioni e comunicazioni in TCP/IP



Uso di **gateway**



Tutti i servizi applicativi a livello utente in UNIX

sono organizzati al disopra di

IP forwarding di **pacchetti**

ICMP forwarding di **pacchetti di controllo**

UDP servizi senza **stato**

TCP garanzie di **correttezza**
connessioni bidirezionali
controllo di flusso
dati out-of-band

Livelli e nomi dei relativi dati manipolati

LIVELLO	TIPO DATI IN INGRESSO	TIPO DATI IN USCITA (sotto)
Applicazione	*messaggio* utente	*messaggio*
Trasporto	*messaggio*	*segmento pacchetto*
IP	segmento visto come *pacchetti*	*datagramma*
Interfaccia fisica di rete	datagramma	*frame* fisico

TCP (trasporto)

Servizio logico
Trasmissione di messaggi con caratteristiche
- **connessione (senza qualità) e non connessione**

in caso, di CONNESSIONE
- **connessione bidirezionale**
- **dati differenziati (normali e prioritari)**
- **controllo flusso byte**
  ordine corretto dei byte,
  ritrasmissione messaggi persi
- **controllo di flusso**
  bufferizzazione
- **multiplexing**
- **semantica at-most-once** (non exactly-once)
  che consenta durata limitata e **di avere eccezioni**
  nel modo più trasparente possibile

IP (Rete)

- Problema dei ***nomi***
  astrazioni *=> spazio dei nomi*

- ***Protezione*** delle informazioni
  astrazioni => spazi di nomi gerarchici

- Routing (trasparente?)
  a livello di rete (e non di nodo)

altri sistemi di nome: NOMI DINAMICI LOGICI

INDIRIZZAMENTO GERARCHICO

a livello di IP

Ogni connessione di un host a una rete ha un indirizzo internet unico di 32 bit

IP-ADDRESS {**NETID, HOSTID**}
un identificatore di rete NETID e
un identificatore di host HOSTID

La distinzione facilita il routing

Legame con la rete e routing
ip individua connessioni nella rete virtuale
==> astrazione dell'indirizzo hardware fisico
indipendente da questo
(**non in dipendenza dalla locazione di accesso**)

- host con più connessioni hanno **più indirizzi** (multiporta per bridge o gateway)
- se un host si collega in una rete diversa **deve** cambiare il suo **ip**, in particolare il **netid**, e può mantenere il proprio **hostid**
  se un host nella stessa rete usa una connessione diversa, **può** cambiare il suo **ip**, in particolare **hostid**

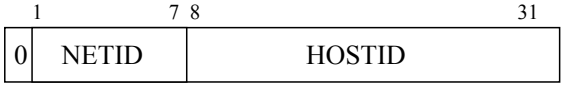
STANDARD
nomi dati di autorità
Network Information Center (NIC) assegna il numero di rete, cioè l'informazione usata nei gateway per routing

NOMI di NODI

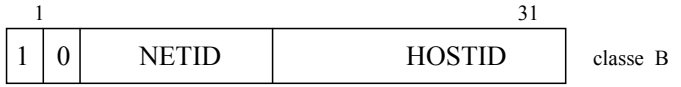
a livello di IP

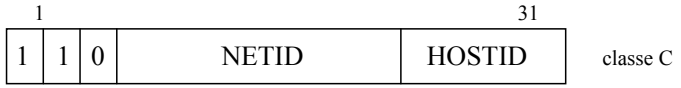
Gli indirizzi sono suddivisi in **3 classi** primarie
(in base al numero di reti e al numero di host collegabili)
e differiscono per il numero di bit assegnati ai singoli identificatori:

Le WAN hanno generalmente **ip-address** di **classe A**



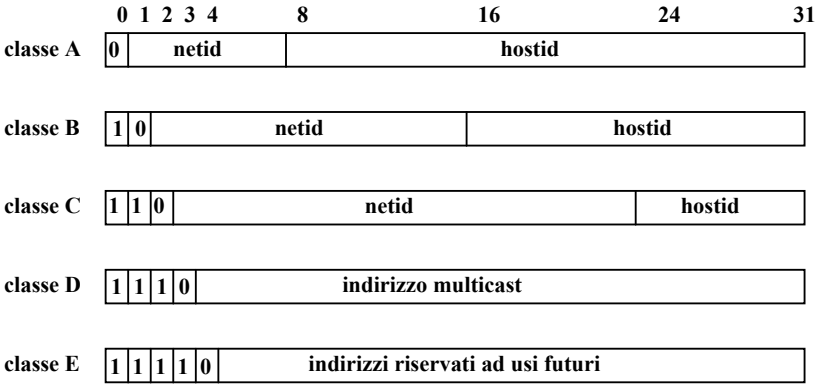
Le LAN hanno **ip-address** di **classe B** o **C**





analizzando un indirizzo IP si può distinguere la classe
in modo automatico

CLASSI di indirizzi



Classi degli IP address

Azioni di gruppo

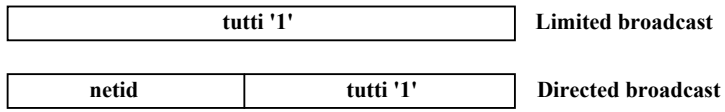
Indirizzamenti broadcast

tutti gli host della rete locale indipendentemente dall'indirizzo IP ==> indirizzo in cui tutti i 32 bit sono a 1 (***limited broadcast*** *address*) **intranet**

non viene fatto passare da una rete ad un'altra

tutti gli hosts in una rete specifica ==> tutti i bit di hostid a uno (***broadcast direttivo*** *o directed broadcast*) **internet**

arrivato alla destinazione, broadcast



Indirizzamenti multicast di Classe D

tutti gli host che si sono registrati possono ricevere (?)

SISTEMI di NOMI IP per i NODI

Ogni **protocollo** deve definire i propri **nomi**
Indirizzi Internet
Un nodo è qualificato come **Rete** e **Host**
Potenzialità di numeri elevati di nodi distinti: **32** bit
Tre classi di indirizzi fisici (a byte) **Network** e **Host**

classe A:	Network	Host	
	0 7 bit	24 bit	
1.#.#.#	**126**.#.#.#	127 riservato per usi locali	
arpa 10			

classe B:	Network	Host
	10 14 bit	16 bit
128.0.#.#	**191.255**.#.#	

almanet	**137.204.#.#**
cineca	130.186.0.0
deis33	137.204.57.33
dida01	137.204.56.1
didasun1	137.204.56.20
hp735	137.204.58.42

classe C:	Network	Host
	110 21 bit	8 bit
192.0.0.#	**223.255.255**.#	
cnrbologna	192.94.70.0	

classe D:	**224**.#.#.#	**239**.#.#.#

Indirizzi in più forme (sintattiche)

Più sistemi di nomi

forme fisiche

10001001 00001010 00000010 00011110

scritti usualmente nella forma più leggibile

dot notation **137.10.2.30**

Risoluzione degli indirizzi (dal livello N al D)

Due macchine che comunicano hanno

indirizzi fisici: F_a, F_b *(DATA LINK)*

indirizzi di IP: I_a, I_b *(RETE)*

problema della risoluzione dell'indirizzo

due modi principali

mappaggio diretto

associazione dinamica

mappaggio diretto per piccole reti

scelta indirizzo hardware per ogni macchina

la risoluzione dell'indirizzo consiste nella sola estrazione del nome fisico dall'indirizzo IP

In reti più grandi ==> risoluzione più complicata

ETHERNET prevede un indirizzo fisico di 48 bit assegnato alla scheda di interfaccia

Questo indirizzo non può essere tradotto nei 32 bit del formato degli indirizzi IP

Traduzione dell'ip-address in indirizzo fisico:

necessità di un protocollo dinamico

Vantaggio del naming di TCP/IP

possibilità di utilizzare indirizzi aventi la stessa forma per riferirsi:

a un host (netid,hostid);

a una rete (netid,0);

a tutti gli host su una rete, broadcast (netid,1...1).

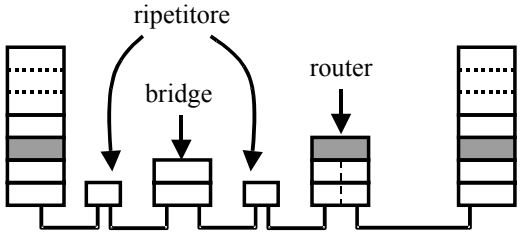
La sottostante tecnologia di rete determina l'effettiva possibilità e efficienza delle trasmissioni broadcast

Per consentire lo scambio dei dati binari (in particolare degli indirizzi) i protocolli TCP/IP hanno fissato uno standard universale per l'ordine dei byte nella codifica dei numeri interi: il primo byte è il più significativo

Big Endian vs Little Endian

SUN, HP big-endian

Intel little-endian



I **router** con nomi IP

i livelli sottostanti sono invisibili:

i **ripetitori** e i **bridge** non hanno nomi IP

ARP (Address Resolution Protocol)

Ricerca dell'indirizzo fisico di un nodo

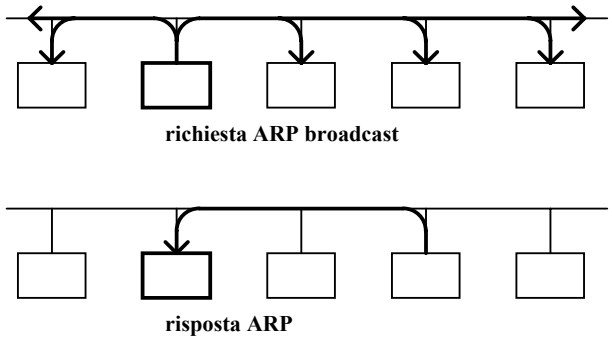
ARP protocollo

semplice ed efficiente (costo broadcast)

invia un pacchetto broadcast in cui chiede l'indirizzo fisico corrispondente ad indirizzo IP
(Quale F_a per questo I_a?)

tutti gli hosts ricevono tale pacchetto

solo quello che riconosce il **suo indirizzo IP** risponde con il proprio indirizzo fisico



Schema di funzionamento protocollo ARP

Questo meccanismo non viene attivato per ogni pacchetto

Utilizza di una **memoria cache** per mantenere le associazioni {indirizzo IP-indirizzo fisico} già usate
cache consultata prima di usare ARP

soft state: *lo stato viene mantenuto per un certo tempo, poi scade naturalmente e deve essere rinnovato*

ottimizzazioni:

- l'associazione relativa alla macchina richiedente memorizzata anche dalla macchina che risponde ad ARP
- ogni richiesta broadcast viene memorizzata da tutti
- una nuova macchina al collegamento invia sulla rete locale un broadcast con la propria coppia
  {indirizzo fisico - indirizzo IP}

ARP distingue **due ruoli** nel protocollo

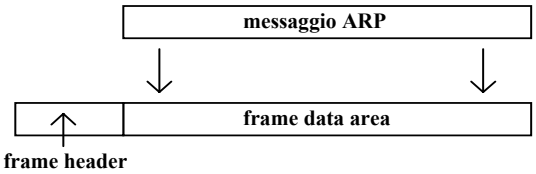
una **attiva** determina l'indirizzo fisico per ogni pacchetto
una **passiva** risponde alle richieste delle altre macchine

Attivo esamina la cache per risolvere indirizzo IP locale
altrimenti esegue una richiesta ARP broadcast (cliente)

la gestione della *richiesta broadcast* deve prevedere di non ricevere risposta o riceverla con ritardo

Passiva risponde alle richieste di altri (server)
estrae sia indirizzo IP sia il fisico per un pacchetto ARP
controlla che non esista in cache e processa il pacchetto
Se risoluzione del proprio indirizzo ==> invio risposta

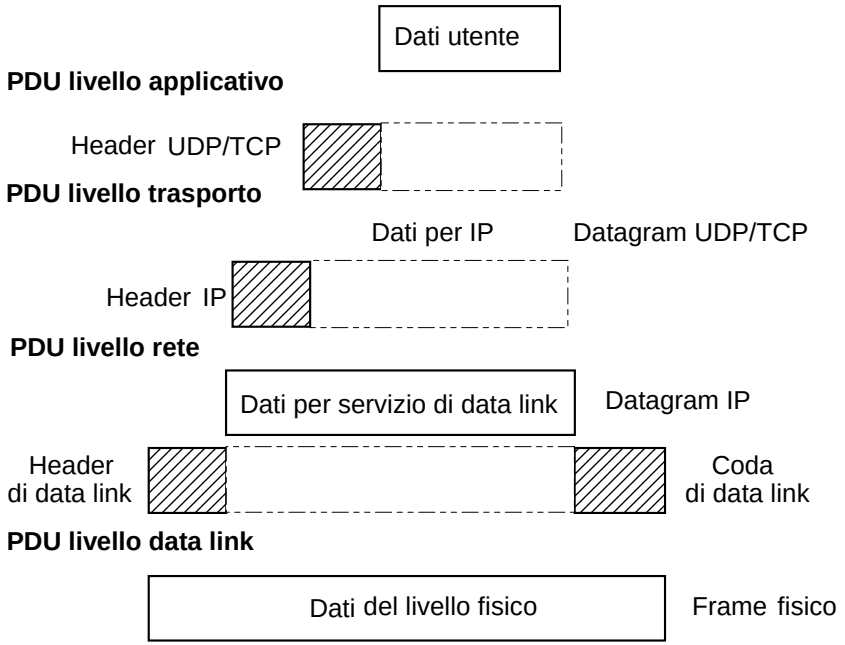
Un messaggio ARP incapsulato in frame fisici e reinviato al richiedente



vedi comando **arp**

Passaggio delle informazioni

tra i diversi livelli nella suite (UDP/)TCP / IP



Ogni PDU formato ad un livello diventa il SDU per il livello inferiore

Si noti che il livello di DATA LINK aggiunge un footer

Un router intermedio può cambiare il datagramma?

FORMATO di un frame ETHERNET

campi in byte

PREAMBOLO	7	10101010
delimitatore di inizio frame (Start Frame Delimiter)	1	11010101
DESTINATION address	2/6	
SOURCE address	2/6	
type (id protocollo ARP/RARP ...)	2	
DATA . . .	46.. 1500	
controllo di fine frame (Frame Check Sequence)	4	

Dati nel frame da 46 a 1500 ottetti
(lunghezza senza preambolo da 46 a 1518)

Anche gli altri livelli MAC introducono forma analoghe per i frame corretti

In genere:
- indirizzi a 48 bit per il nodo mittente e destinatario
- si introducono sia preamboli, sia delimitatori finali
- controllo del frame attuato con controllo CRC

In un frame per **1 solo byte** (escluso preambolo e CRC)
con 1 byte applicativo ==> overhead 46 byte
20 IP e **20** TCP/UDP
15 riempimento

Formato header ARP/RARP

Le informazioni di protocollo sono inserite in un frame di livello data link

campi in byte

PREAMBOLO	7	10101010
delimitatore di inizio frame (Start Frame Delimiter)	1	11010101
DESTINATION address	6	
SOURCE address	6	
type (id protocollo ARP/RARP ...)	2	
DATI del Protocollo ...	**vedi**	
controllo di fine frame (Frame Check Sequence)	4	

Si possono considerare i due protocolli insieme

0 15

Tipo Hardware	
Tipo protocollo	
Lunghezza nome Hw	**Lunghezza nome IP**
operazione: 1 - 4	
Indirizzo Sender Hw	
Indirizzo Sender IP	
Indirizzo Receiver Hw	
Indirizzo Receiver IP	

operazione: 1 ARP request 2 ARP response

operazione: 3 RARP request 4 RARP response

protocollo RARP

(*Reverse Address Resolution Protocol*)

Ricerca indirizzo IP di un nodo

Indirizzo IP in memoria secondaria
che il sistema operativo cerca allo startup

e macchine diskless?

indirizzo IP viene ottenuto richiedendolo ad un server
Assumiamo che tale server possieda un disco in cui siano contenuti gli indirizzi internet

Si usa provvisoriamente l'indirizzo fisico
indirizzo fisico è fornito dall'interfaccia di rete hardware

protocollo RARP (Reverse ARP) di basso livello
Uso diretto della rete fisica ==>
il protocollo RARP gestisce la ritrasmissione e la perdita di messaggi

Cliente
Uso di **broadcast** per conoscere il proprio indirizzo IP
e se non c'è risposta? ritrasmissione

Servitore
invia la risposta a chi ne ha fatto richiesta

Si **prevedono più server per ogni LAN**
per rispondere ai clienti anche in caso di guasto

Server multipli

Modello a server attivi
Troppi server sovraccaricano il sistema se cercano di rispondere contemporaneamente alla richiesta

Modello a server attivi/passivi
soluzioni possibili con **gerarchia** di server
(tipicamente 2 server)

Modello dinamico con server passivi in ascolto
si prevede
Il server **primario** è il solo a rispondere
gli altri server rispondono solo se arriva una seconda richiesta RARP

Modello statico con server differenziati
(ritardi diversi)
questa soluzione prevede che
il server primario risponda immediatamente
gli altri solo se il primario non risponde e con un ritardo calcolato **random**
la probabilità di risposta simultanea è bassa

Notiamo che consideriamo protocolli che si basano
- **su responsabilità distribuita**
- **su gestori (anche replicati)**

I servizi ottenuti con l'ausilio di servitori cominciano a dare una idea di insieme di servizi e di infrastruttura attiva di gestione

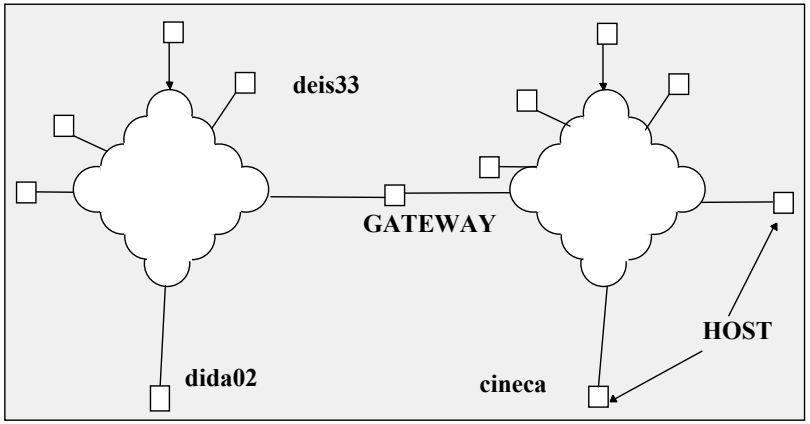
SEPARAZIONE tra reti
RETI uniche logicamente connesse
RETI fisiche separate

SI INDIRIZZA in modo **diretto** solo
NELL'AMBITO DELLA **STESSA RETE**

per esempio
dida01 e *deis33* sono in grado di comunicare
perchè sono in classe B e nella stessa rete

dida02 e *cineca* non possono comunicare direttamente

In casi indiretti ogni comunicazione richiede un intervento di un gateway autorizzato (**router**)



Il gateway deve comparire con (almeno) 2 indirizzi, uno su ciascuna rete su cui si affaccia

Ulteriore protezione SUBNETTING

Sottoreti (politica locale)

Una rete può essere divisa in **sottoreti** al suo interno *(all'esterno non è visibile la suddivisione)*
le rete stessa rispetta **localmente** la maggiore **granularità**

subnet ==> il campo **host** è ulteriormente suddiviso

		subnet	*host*
		8 bit	8 bit
dida01	137.204.56	subnet 56	
deis33	137.204.57	subnet 57	

La **sottorete** è rispettata **comunicando direttamente** solo nella sottorete stessa, altrimenti tramite **gateway**

meccanismo ==> **maschere** di *chiusura* e *protezione*
11111111 11111111 11111111 0000000
NETMASK qui maschera in classe B (per 3 byte)
255.255.255.000

MASCHERA come bit di rete impedisce di uscire FUORI dalla SOTTORETE
La decisione di mascherare è locale ad ogni connessione
e si potrebbe anche non rispettare
deis33 non subnet ==> comunicazione diretta con gli host della stessa **network e subnetwork** (ignorando subnet)

DALL'ESTERNO DELLA RETE

nessuna differenza

ALL'INTERNO DELLA RETE

quando il messaggio è arrrivato un accordo tra i gateway renda attiva la suddivisione, usando un servizio di **routing** per portare il messaggio alla corretta sottorete e, di lì, alla destinazione

coordinamento di
tabelle di routing

per *deis32-35*, cioè su *deislan*
si devono individuare i router per le altre sottoreti

	network	**gateway di routing**
cineca	default	137.204.57.253
didalan	137.204.56	137.204.57.33
deislan	137.204.57	137.204.57.33
cciblan	137.204.58	137.204.57.33

Il subnetting rende possibili ulteriori suddivisioni dello spazio dei nomi IP (non deducibili automaticamente dal nome IP)

NOMI FISICI IP

Ma bisogna **sempre** usare i nomi fisici?
Possibilità di nomi ***logici più significativi***
mantenendo la stessa protezione $\Rightarrow$ uso di DNS

INTERNET PROTOCOL
IPv4 - Datagrammi senza connessione

1 **SERVIZIO**

- **connectionless**: ciascun pacchetto è trattato indipendentemente dagli altri. Diversi pacchetti possono seguire percorsi diversi ed essere consegnati fuori ordine
- **unreliable**: la consegna non è garantita, cioè non effettua un controllo sull'avvenuta ricezione di un pacchetto
- **best-effort**: l'inaffidabilità del trasferimento è dovuta a cause esterne e non al software di rete nessun messaggio di errore al richiedente

2 **PROTOCOLLO** due funzioni principali

- **elaborazione** del messaggio del livello superiore nel formato per la trasmissione
  - incapsulamento / frammentazione
- **instradamento (routing)** cioè:
  - traduzione da indirizzo logico a indirizzo fisico;
  - scelta del percorso

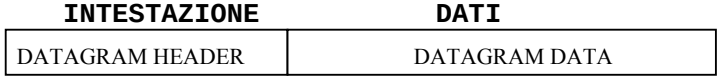
3. **REGOLE**

- formato del **datagramma**, unità base di informazione da trasmettere
- la **sequenza di operazioni** che deve essere eseguita per effettuare una comunicazione
- la **gestione degli errori** (molto limitata: in genere eliminazione del datagramma)

IP-DATAGRAM

Unità base di informazione che viaggia in Internet

Suddiviso in due parti principali:



IP non specifica il formato dell'area dati
dati di qualunque tipo

Formato dell'IP-DATAGRAM

I sottocampi del campo header contengono:

- ***versione*** del protocollo
- ***lunghezza*** header e totale (*totale < 64K*)
- ***identificazione*** del datagramma (usato per ricomporre i frammenti)
- ***precedenza*** (0-7)
- ***tipo di trasporto*** desiderato (bit di qualità)
  Type **o**f **S**ervice (**ToS** vedi **qualità del servizio**)
  throughput **T**,di affidabilità **R**, di ritardo **D**, costo **C**

Internet non può garantire il soddisfacimento del tipo di trasporto richiesto che dipende dal cammino che deve percorrere il datagramma

- ***frammentazione*** e ***flags***
- ***time to live***, tempo di permanenza del datagramma
- ***indirizzo IP*** sorgente e destinazione
- **tipo di protocollo *protocol*** (TCP 6, UDP 17, ICMP 1, ...)
- **checksum** per il controllo
- ***opzioni: monitoraggio*** e ***controllo*** rete

Formato dell'Header e Dati

di un datagramma

Header (minimo 20 byte, max 64)

Dati

0	4	8	16	19	24	31
VERS	HLEN	SRV TP	TOTAL LENGHT			
IDENTIFICATION			FLAGS	FRAGMENT OFFSET		
TIME TO LIVE		PROTCL	HEADER CHECKSUM			
SOURCE IP ADDRESS						
DESTINATION IP ADDRESS						
IP OPTIONS (IF ANY)					PADDING	
DATA						
. . .						

0	3	SeRVice TyPe			
PRECEDENCE	D	T	R	C	UNUSED

FLAGS

Do not fragment	More fragments	UNUSED

fragment offset => allineato agli 8 byte (solo 13 bit)

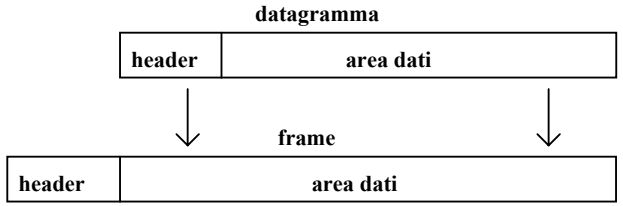
checksum => somma complemento a 1 delle parole (16 bit alla volta)

frammentazione

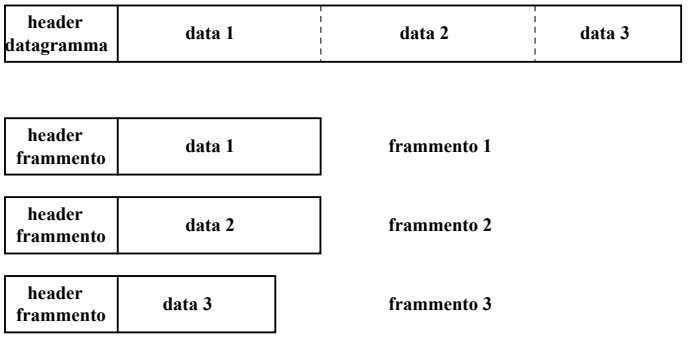
spezzare i datagrammi durante il percorso

- ***DECOMPOSIZIONE*** al mittente
- ***DECOMPOSIZIONE*** ad ogni intermedio
- ***RICOMPOSIZIONE*** al destinatario

datagramma



Incapsulamento datagramma



Frammentazione datagrammi

frammentazione

I datagrammi devono essere incapsulati nei frame di livello 2 delle reti fisiche su cui transitano

MTU (maximum transfer unit)

lunghezza massima dei frames a livello fisico

dimensione massima del datagramma

1ª possibilità: ***calcolo statico da parte del mittente***

il datagramma incapsulato nel singolo frame fisico (dimensioni del datagramma minore o uguale alla ***più piccola MTU*** presente in Internet)

a livello utente trasmissione con tempi molto lunghi per il trasferimento di un messaggio (se MTU molto piccole)

efficiente solo per reti fisiche con MTU a lunghezza elevata ed omogenea

*2ª possibilità (**USATA**)*

MTU scelta indipendente dalle tecnologie sottostanti per rendere efficiente la comunicazione a livello utente (fissata tipicamente a 64Kbyte) ==>

Il pacchetto originale viene suddiviso in **frammenti** su MTU a dimensione inferiore (a 64Kbyte)

La frammentazione del pacchetto può avvenire ad ogni passo nelle reti intermedie e si richiede riassemblaggio al destinatario

OPZIONI: Monitoraggio e controllo rete

Le opzioni più interessanti sono:

record route genera una lista degli inidirizzi IP dei gateway che il frame ha attraversato (***al massimo 9***)

otteniamo una indicazione dei gateway intermedi

timestamp genera una lista dei tempi di attraversamento degli intermedi

possiamo ottenere una indicazione della permanenza nei gateway intermedi (vedi mail)

source route il sorgente fornisce indicazioni sul cammino da seguire nel routing del frame
instradamento al sorgente

si dirige il cammino dal sorgente

- ***strict source***: una indicazione di tutti i gateway intermedi da attraversare
- ***loose source***: una indicazione di un insieme di percorsi da attraversare

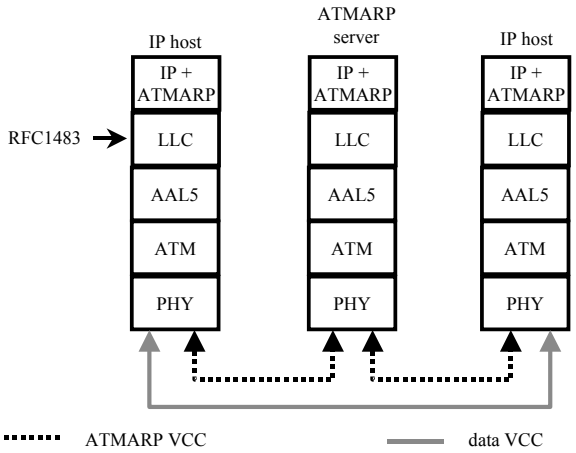
Numero massimo di informazione nel datagramma:

limite al controllo del percorso (9 passi)
uso di area opzioni (44 byte) 11 parole
1 codice, 1 contatore, 9 informazioni ripetute

security
stream identification

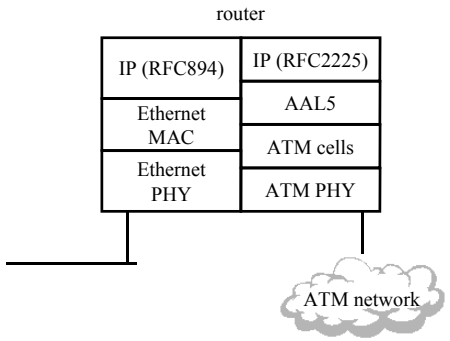
IP su ATM

Il protocollo IP può anche usare altri protocolli per il trasporto tra reti: ad esempio tra dorsali si può sfruttare ATM



AAL5 esegue la frammentazione e la gestione celle a default

MTU 9180 bytes che può essere cambiato



IP-ROUTING

controllo della trasmissione messaggi

Astrazione

- si ragiona in termini di **reti**
- si ragiona per interconnessioni limitate di **reti**

In Internet host e gateway possono partecipare entrambi al routing, ma con regole diverse

INTERNET ***instradamento attraverso diverse reti***
INTRANET ***algoritmo di routing all'interno di una rete***
dipende dalla tecnologia sottostante

IP-ROUTING

Routing DIRETTO e INDIRETTO

1. **Direct routing (INTRANET o SUBNET)**
   host o gateway che invia il messaggio a un host che si trova sulla stessa rete fisica

Operazioni svolte:

- il datagramma viene incapsulato in un frame fisico
- viene effettuata la traduzione da ip a indirizzo fisico
- trasmissione dal mittente al destinatario

2. **Indirect routing (INTERNET)**
   host mittente e destinatario connessi su reti diverse

Il datagramma passa da un gateway ad un altro fino ad un gateway che può inoltrarlo direttamente
Alterata solo la parte di frame fisico (checksum e fram.)
Utilizzo di ***tabella di routing*** che lavora (per lo più) sulle informazioni di rete

Algoritmi di routing

Algoritmi generalmente globali basati su tabelle che sono disponibili ai diversi router partecipanti
A parte alcuni casi iniziali:
uso di protocolli isolati tipo patata bollente

1° possibilità: STATICO
basato su **informazioni statiche** riguardanti il cammino più breve (per piccole reti ed interconnessioni)

2° possibilità: DINAMICO
basato su **informazioni dinamiche** di traffico della rete, lunghezza del messaggio e tipo di servizio richiesto

Router che si coordinano attraverso protocolli

algoritmi GLOBALI (per le tabelle) e DISTRIBUITI

- **time out** delle entry dei router in modo asincrono
- propagazione **asincrona** delle informazioni di routing

in INTERNET adatti per configurazioni STATICHE

Distance Vector
ogni gateway mantiene la distanza di ogni altra rete (in genere in hop) e il vicino attraverso cui instradare

Link State (shortest path)
ogni gatweway ha le informazioni di tutto il sistema

*Problemi in caso di **variazioni dinamiche** delle tabelle*

Algoritmo **Distance Vector**

Tabelle di routing per ogni gateway senza conoscenza completa del cammino di interconnessione ma con informazioni globali

Definizione di una metrica: ad esempio numero dei passi per raggiungere una rete
In ogni gateway NON si mantengono i cammini completi, ma solo del primo passo e della distanza

FASE di PROPAGAZIONE

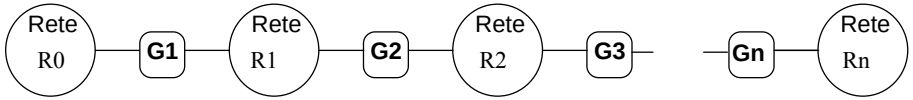


Tabelle al primo passo

R0 0
R1 0

R1 0
R2 0

R2 0
R3 0

...

Rn-1 0
Rn 0

Tabelle al secondo scambio

R0 0
R1 0
R2 1 G2

R1 0
R2 0
R0 1 G1
R3 1 G3

R2 0
R3 0
R1 1 G2
R4 1 G4

...

Rn 0
Rn-1 0
Rn-2 1 Gn-1

A regime, ogni gateway contiene la distanza di ogni rete

G1
R0 0
R1 0
R2 1 G2
R3 2 G2

G2
R1 0
R2 0
R0 1 G1
R3 1 G3

G3
R2 0
R3 0
R1 1 G2
R4 1 G4

...

Gn
Rn 0
Rn-1 0
Rn-2 1 Gn-1
Rn-3 2 Gn-1

Caso di Propagazione ... Lenta

G1
R0 0
R1 0
R2 1 G2
R3 2 G2

G2
R1 0
R2 0
R0 1 G1
R3 1 G3
R4 2 G3

G3
R2 0
R3 0
R1 1 G2
R4 1 G4
R0 2 G2
R5 2 G4

...

Gn
Rn 0
Rn-1 0
Rn-2 1 Gn-1
Rn-3 2 Gn-1
Rn-4 3 Gn-1
Rn-5 4 Gn-1

G1
R0 0
R1 0
R2 1 G2
R3 2 G2
R4 3 G2
R5 4 G2

G2
R1 0
R2 0
R0 1 G1
R3 1 G3
R4 2 G3
R5 3 G3
R6 4 G3

G3
R2 0
R3 0
R1 1 G2
R0 2 G2
R4 1 G4
R5 2 G4
R6 3 G4
R7 4 G4

...

Gn
Rn 0
Rn-1 0
Rn-2 1 Gn-1
Rn-3 2 Gn-1
Rn-4 3 Gn-1
Rn-5 4 Gn-1
Rn-6 4 Gn-1

FASE di propagazione molto lenta (*esponenziale nel numero dei nodi*)

G1
R0 0
R1 0
R2 1 G2
R3 2 G2
R4 3 G2
R5 4 G2
...
Rn n-1 G2

G2
R0 1 G1
R1 0
R2 0
R3 1 G3
R4 2 G3
R5 3 G3
R6 4 G3
...
Rn n-2 G3

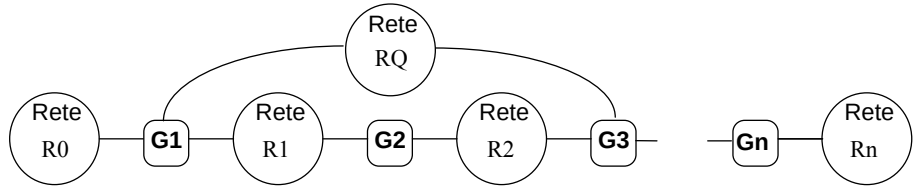
G3
R0 2 G2
R1 1 G2
R2 0
R3 0
R4 1 G4
R5 2 G4
R6 3 G4
R7 4 G4
...
Rn n-3 G4

...

Gn
R0 n-1 Gn-1
R1 n-2 Gn-1
...
Rn-6 4 Gn-1
Rn-5 4 Gn-1
Rn-4 3 Gn-1
Rn-3 2 Gn-1
Rn-2 1 Gn-1
Rn-1 0
Rn 0

Caso di Variazione

Variazione tabelle per una variazione di configurazione



G1
R0 0
R1 0
RQ 0
R2 1 G2
R3 1 G3
...

G2
R1 0
R2 0
R0 1 G1
R3 1 G3
RQ 1 G3
...

G3
R2 0
R3 0
RQ 0
R1 1 G2
R0 1 G1
...

Gn
Rn 0
Rn-1 0
Rn-2 1 Gn-1
Rn-3 2 Gn-1
Rn-4 3 Gn-1
...

Propagazione locale delle **tabelle di routing** ad ogni vicino in modo asincrono

Chi riceve una offerta aggiorna la propria tabella se la proposta è conveniente in base alla metrica

Le entry hanno scadenza (e devono essere sostituite)

Ogni gateway decide il routing in modo indipendente in base alla tabella locale

CAMBIAMENTO o VARIAZIONI

in caso di **crash** o di aggiunta di un **nuovo gateway**

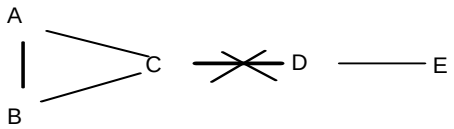
Possibilità di problemi (**cicli**) e non convergenza

SVANTAGGI

- messaggi di aggiornamento hanno una propagazione lunga e convergenza lenta (scambio di tabelle tra vicini)
- i messaggi seguono gli stessi cammini (*tutto il traffico usa un unico cammino, senza bilanciare*)

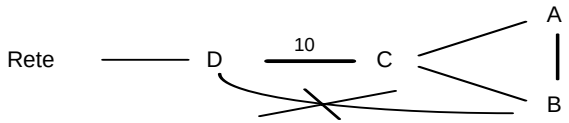
Problemi in Riconfigurazione

prendiamo una configurazione



In caso di guasto del link da D verso C, E ha un valore precedente e lo manda a D che lo instaura, alla scadenza ottiene il valore da D ed incrementa il valore locale, etc.

counting-to-infinity



Riga per le tabelle dopo il guasto di BD per arrivare a Rete

prima Tabelle **dei router dopo** **Tabelle fine**

dir 1	**D**
D 2	**B**
B 3	**C**
B 3	**A**

dir 1
no
B 3
B 3

dir 1
C 4
B 3
C 4

dir 1
C 4
B 5
B 5

...
...
...
...

dir 1
A 12
D 11
B 12

dir 1
C 12
D 11
C 12

A , B e C si danno informazioni sbagliate l'un l'altro

Problema generale dovuto al non tenere traccia di chi fornisce una distanza da un nodo (e cammino relativo)

Spesso si usa la limitazione dell'infinito a 16

Si noti la lenta convergenza del sistema

I messaggi vanno veloci a regime
le cattive notizie si propagano con time-out

Split Horizon

per evitare di passare informazioni sbagliate, non si offrono cammino ai nodi da cui le abbiamo ottenute

C e A non offrono a B che ha dato loro il percorso

Hold-down

dopo una **notifica** di un problema, si ignorano le informazione di cammino per un certo periodo: tutti hanno modo di accorgersi del problema e non ci sono propagazioni errate

loop che si sono già creati vengono mantenuti durante l'hold-down

Split Horizon con poisoned reverse e triggered broadcast

ogni nodo invia immediatamente un broadcast con la indicazione ed il cammino e si usano evoluzioni dello split horizon con conoscenza di cammini (limite di 16)

A invia a C un messaggio di *non raggiungibilità* se crede di raggiungere D via C
C non può rifarsi ad A (che non raggiungeva D)

Ulteriori problemi
altre fasi di broadcast che vengono generate

Evoluzione degli algoritmi per privilegiare variazioni tenendo conto della topologia delle offerte

Algoritmi **Link State**

link-state o Shortest Path First SPF
ogni gateway ha una conoscenza completa della **topologia di interconnessione (grafo completo)**
Tabelle di routing basate sulla conoscenza dell'intero cammino

Il **grafo di interconnessione** per evitare cicli viene gestito con algoritmi che possono favorire decisioni locali (**routing dinamico**)
Dijkstra **shortest-path-first**

Possibilità di fare **source routing** e anche di spedire messaggi su cammini diversi (**routing dinamico**)

A REGIME, ogni gateway tiene sotto controllo le proprie connessioni e le verifica periodicamente

- *invio periodico di messaggi nel vicinato per controllare la correttezza delle risorse locali*
- *identificazione del guasto e segnalazione di eventi di guasto*
  (uso di più messaggi per evitare transitori e accelerare la propagazione)

Non appena si verifica un problema,
chi ha rilevato il problema invia il messaggio a **tutti** i componenti (**broadcast** o **flooding**)

Vantaggi

- si controlla solo il **vicinato**
- informazioni **di variazione** propagate rapidamente
  (senza ambiguità via broadcast)
- possibilità **di scelte differenziate** dei cammini nella topologia
- conoscenza dei **cammini completi** e **source routing**

In sostanza le variazioni non sono dipendenti da possibili intermediari

I messaggi sono gli stessi qualunque sia la dimensione del sistema
SCALABILITÀ

Svantaggi

- *necessità di mantenere tutta la topologia*
- *azioni costose* (broadcast) in caso di variazione

In generale, necessità di limitare i
domini di conoscenza reciproca

Conclusione:
I protocolli globali
non sono scalabili o
sono poco scalabili

ROUTING Architettura INTERNET

NON un insieme di reti collegate direttamente
MA distinzione tra:

Sistemi core e noncore (ARPANET)

core insieme di gateway chiave con informazioni di accesso complete (e replicate)

non core informazioni di routing solo parziali

i nodi CORE si scambiano tutte le informazioni di routing (algoritmo **Distance Vector** e **Link State**)

I problemi sono nati aumentando il numero delle reti pare del sistema ==> astrazione e gerarchia

Sistemi autonomi

insieme di *reti* e *gateway* controllati da una autorità unica centrale, con proprie politiche di routing

I sistemi AUTONOMI devono scambiarsi informazioni di routing e coordinamento solo **intra**sistema.
Il solo gateway di controllo (o i gateway) provvede al protocollo verso l'esterno

Interior **G**ateway **P**rotocol (**IGP**)
protocollo per trovare il percorso all'interno di un sistema autonomo (**intra**sistema)

politica che consente percorsi multipli e con possibilità di tollerare i guasti (algoritmi multipath IGRP CISCO)

Exterior **G**ateway **P**rotocol (**EGP**)
protocollo rilevente per i gateway di controllo per trovare il percorso fino ai core

struttura ad albero con i core come radice

Protocollo di routing per piccole reti

Routing **I**nformation **P**rotocol (**RIP**)
implementato in *routed UNIX*
implementazione di **distance vector**

tutti i nodi partecipano come attivi o passivi
ATTIVI determinano i percorsi
PASSIVI restano ad ascoltare le decisioni degli altri
Le tabelle degli attivi sono significative
le tabelle dei passivi indicano il passaggio attraverso un gateway (attivo)

Ogni 30 secondi si manda un messaggio ai vicini
con la tabella di routing locale

Si aggiornano le tabelle in base ai messaggi ricevuti: se i messaggi rilevano cammini più brevi di quelli noti sono stabiliti i nuovi cammini

Un cammino ha un **time-out** associato e scade dopo un certo intervallo
Ogni nodo viene dichiarato guasto se non ha mandato un messaggio per un **certo intervallo** (180 sec)

☹ **Metrica senza costi di link e valore massimo a 10**
😐 **Capacità di riconfigurazione**

Solo reti di piccole dimensioni

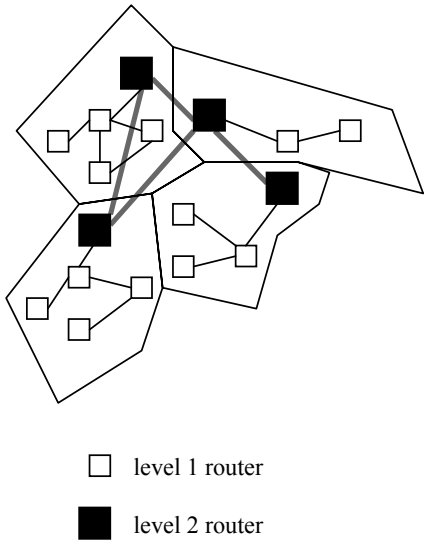
Routing Gerarchico

per aree distinte per gestione
domini amministrativi diversi
unico protocollo di routing per la area

La connessione tra le aree avviene attraverso
gerarchia di router

Routing per livelli

le informazioni di routing possono essere aggregate



Scenario di un area di rete IP

- **autonomous system (AS)**
- **border router e border gateway**

Tipo di traffico

locale	intra-AS
transito	inter-AS

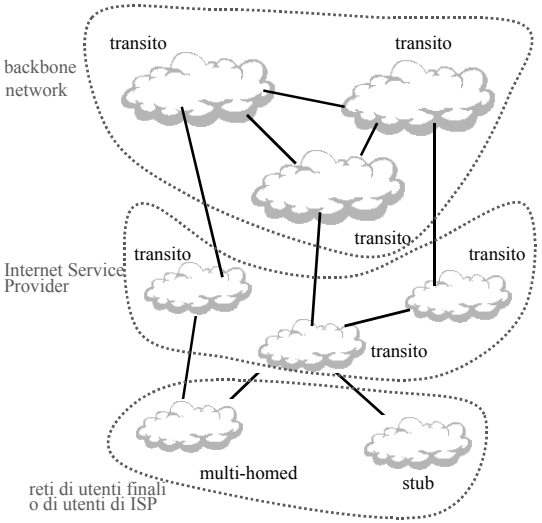
Border Gateway Protocol (RFC 1771)

distance vector ma con cammino
non una metrica ma informazioni di raggiungibilità
possibile un routing basato sulla politica
keep-alive ogni 30s

Autonomous System

ogni AS decide la politica
stub AS e multi-homed AS
AS di transito fungono da backbone provider

Internet



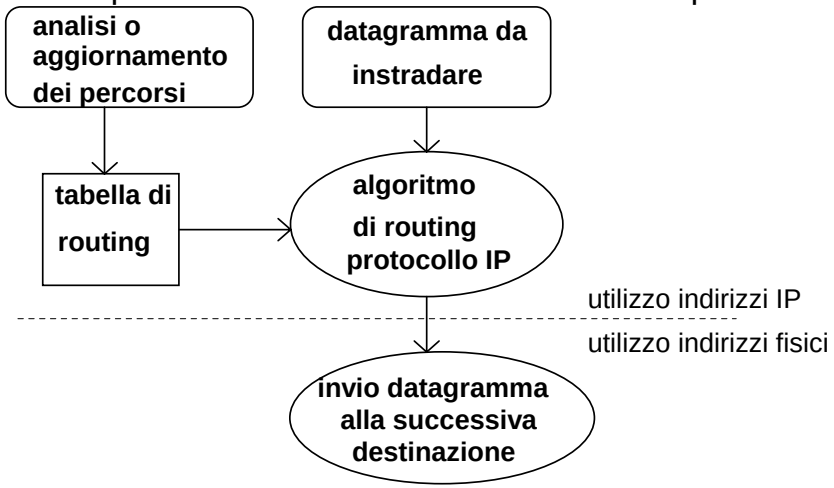
IP_ROUTING non globale

servizio di *instradamento* (routing)
L'IP routing determina l'indirizzo IP del nodo successivo a cui inviare il datagramma
datagramma e indirizzo ==>
e lo passa all'interfaccia di rete

routing con indirizzi IP

decisione del percorso sull'***indirizzo di destinazione*** con tabella di instradamento (Internet Routing Table)
presente sia sugli host che sui gateway
informazioni sulla destinazione e su come raggiungerla

routing IP basato su **informazioni di rete** e **non nodo**
==> ridurre ***dimensioni della tabella*** e
per ***aumentare efficienza*** nella scelta percorso



Software IP e tabella di indirizzamento

si veda **netstat -r**

ALGORITMO DI ROUTING IP

funzione
Route_IP_Datagram (datagram, routing_table)

Separazione indirizzo IP destinatario (***Idest***) datagramma
Valutazione indirizzo IP della rete di destinazione (***Inet***)

if ***Inet*** un indirizzo raggiungibile direttamente
then invio del datagramma alla rete destinataria
(trasformazione indirizzo IP in indirizzo fisico e incapsulamento del datagramma in frame)

else *if* ***Idest*** un host con un cammino proprio
then invio del datagramma in base alla tabella

else *if* **Inet** si può ottenere da una entry nella tabella di routing
(tenendo conto di subnet)
then si invia il datagramma al prossimo gateway

else percorso di default per tutti i restanti datagrammi

Si tiene conto della sottorete
usando la **maschera** ed **Idest**

Si deve anche trattare la possibilità di errori di indirizzamento: ad esempio un host non esistente sulla rete locale

DATAGRAMMI in ingresso
host o gateway tratta i datagrammi ricevuti

router livello IP verifica se destinatario utente locale
if arrivato,
then lo accetta e lo passa al protocollo di alto livello
else controllo destinazione
evitando rinvii di datagrammi consegnati per errore

gateway due casi: destinazione finale o altro invio
instradamento con algoritmo standard usando informazioni della tabella di indirizzamento locale

Problemi
- host con più connessioni fisiche e più indirizzi IP
- datagrammi broadcast
  anche decremento del campo 'time_to_live' nel datagramma, scarto del datagramma se zero

Routing Information Protocol (RIP) (RFC1058)
ispirato a **distance vector** (con modifiche) basato su
- ruoli attivi e passivi
- broadcast (30 secondi) di messaggi di cambiamento
- mantiene vecchi cammini
- elimina problemi di non convergenza (infinito a 16)

(con split horizon triggered update poisoned reverse)

RIPv2 (RFC2453)
- informazioni addizionali come {dominio routing e tag}
- anche IP address mask per indirizzi classless
- si prevede autenticazione semplice

Open Shortest Path First (RFC2328)
La comunità Internet si è adeguata ad un protocollo di tipo link state

Open SPF Protocol
(link state o Shortest Path First)

con servizi ulteriori
- cammini multipli e load balancing, cammini specifici
- introduzione di aree auto-contenute
- autenticazione
- definizione di risorse virtuali
- ottimizzazione delle risorse (broadcast)

☹ maggiori costi computazionali del RIPv2:
i router devono mantenere l'intero albero
☹ più difficile da implementare del RIPv2(?)
☺ cambiamenti via multicast:
non broadcast/flooding, carico basso
☺ maggiore stabilità del RIPv2:
convergenza veloce

☺ si possono usare fattori di QoS
☺ bilanciamento del carico
cammini multipli per la stessa destinazione
☺ si possono fare autenticazioni

Internet Protocol v6 (IPv6)

a fronte dell'esaurimento degli indirizzi IP
2,11 M reti (alcune classi C libere), 3,72 G connessioni
nuove proposte di sistemi di routing e di nomi

IPv6 => 128 bit / 16 byte
forte estensione del sistema
(7 10^{23} indirizzi per metro^2)
mantenendo anche la compatibilità con IPv4

X:X:X:X:X:X:X:X dove X sta per una word a 16 bit
0:0:0:0:0:0:137.204.57.33 o ::137.204.57.33

La scelta è nata dopo discussioni e varie proposte

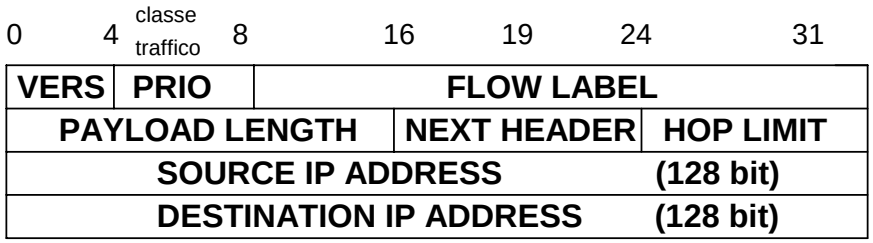
Gerarchia di indirizzi divisi per
forniture di servizi e indirizzi geografici,
usi locali e non visibili
Inoltre, si riconoscono anche funzionalità

- **point-to-point**
- **multicast**
- **anycast** (un insieme di destinatari di cui si deve raggiungere il più vicino o comodo)

0: **IPv4**
1: OSI
2: Novell
...
255: **Multicast**

IPv6

L'header del messaggio è più **limitato e fisso**
senza variazioni (8 byte)
a parte gli indirizzi del ***mittente e destinatario***
solo in caso di necessità si punta ad header di estensione



PRIO Type of Service
0-7 best effort 8-15 Streaming e QoS

FLOW LABEL 24 bit
per tenere traccia di flussi da trattare nei diversi cammini

PAYLOAD
minima 536 massima 64K

NEXT HEADER (**type length value**)
uso di estensioni segnalati con header aggiunti
hop by hop
routing
fragment
authentication
encapsulating security payload
destination options

HOP LIMIT
tipo il time to live (IPv4)

Protocollo ICMP

Internet Control Message Protocol (ICMP)
Gestione della rete (ed errori)
Controllo della rete

ICMP consente di inviare messaggi di **controllo** o di **errore**
al **sorgente** del messaggio (solo a questo)
ICMP usato per il coordinamento tra livelli di IP

Condizioni di errore al mittente (non correzione)
per i relativi provvedimenti
nodi intermedi non informati dei problemi
nodo sorgente può provvedere a correggere

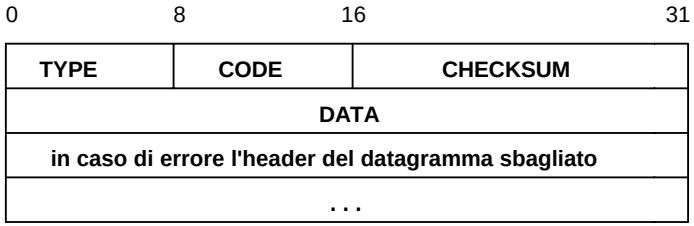
ICMP
Rappresenta un mezzo per rendere note condizioni anomale a chi ha mandato datagrammi (usando IP)
La politica di uso è tutta a carico dell'utilizzatore

METALIVELLO
e gli errori sugli errori?
Errori su messaggi ICMP non possono causare a loro volta messaggi ICMP

I messaggi ICMP sono considerati a livello di datagrammi IP
sono soggetti alle stesse regole di routing
- non hanno priorità
- possono essere persi
- possono causare ulteriore congestione

FORMATO

type		identificatore del messaggio
code		informazioni sul tipo di messaggio
checksum	(16 bit)	utilizzato dal relativo algoritmo



Messaggio ICMP inserito un datagramma IP: il messaggio ICMP contiene sempre l'header e 64 bit dell'area dati del datagramma che ha causato il problema

I campi type e code consentono di fornire informazioni ulteriori

Possibili valori del campo type

0	*Echo Reply*
3	**Destinazione irraggiungibile**
4	**Problemi di congestione (*source quench*)**
5	Cambio percorso (*redirect*)
8	*Echo Request*
11	**Superati i limiti di tempo del datagramma**
12	**Problemi sui parametri del datagramma**
13	Richiesta di timestamp
14	Risposta di timestamp
15	Richiesta di Address mask
16	Risposta di Address mask

Eventi segnalati

campo *CODE* ==> un intero dipendente dai valori di *TYPE*
Se il destinatario non si raggiunge
campo type vale 3 e *campo code* codice di errore

0	Rete irraggiungibile
1	Host irraggiungibile
2	Protocollo irraggiungibile
3	Porta irraggiungibile
4	Frammentazione necessaria
5	Errore nel percorso sorgente (source route fail)
6	Rete di destinazione sconosciuta

ICMP livello errori

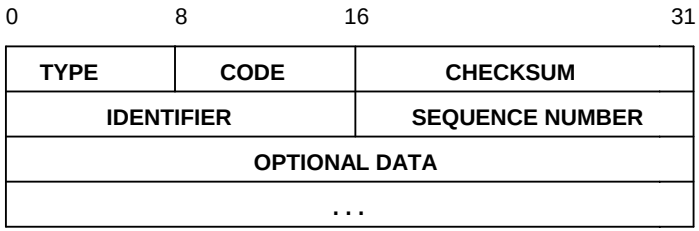
***destination unreachable* (type 3)**
Network unreachable (code 0)
Frammentazione necessaria, ma non consentita
Route a sorgente non esatta (*source route failed*)

***source quench* (type 4)** *caso di congestione*
Se il buffer dedicato ai frammenti e datagrammi è esaurito, sono scartati: si invia un avvertimento al mittente

***cicli e perdita di datagrammi* (type 11)**
problemi su un *datagramma singolo*
scadenza del *time-to-live* o del *tempo di ricomposizione*

ICMP livello coordinamento

Invio di informazioni di routing tra gateway



***echo request/reply* (type 8/0)** *controllo percorso*
un host verificare la raggiungibilità di una destinazione
Per esempio:
- si può verificare che un host esista
  inviando un **echo request (type 8)**
  ricezione di **echo request (type 0)**

(**ping** : echo request al nodo e attesa di echo reply stimando il RTT)

Altri messaggi di controllo

***address mask* (type 17/18)** *richiesta di maschera*
un gateway deve conoscere una sottorete

***sincronizzazione degli orologi* (type 13/14)**
ricezione e invio del tempo fisico
si misurano i millisecondi
si considera tempo di invio, di ricezione, di risposta

***redirect* (type 5)** *cambio percorso*
un gateway deve cambiare la propria tabella di routing
funzione di controllo di gestione

Comando traceroute
che visualizza il percorso fino ad un nodo

Si mandano messaggi con TTL crescente
il nodo che arriva a TTL=0 scarta e
manda un messaggio ICMP
Ogni perdita forza un messaggio ICMP
che viene catturato al mittente

assumiamo che nel frattempo non cambino le tabelle di routing:
altrimenti potremmo avere dei problemi di inconsistenza

Comando ping
Si manda un echo request al nodo
si aspetta un echo reply
si stima il tempo RTT

si possono variare le dimensioni dei dati ed il numero di invii
secondo le diverse necessità

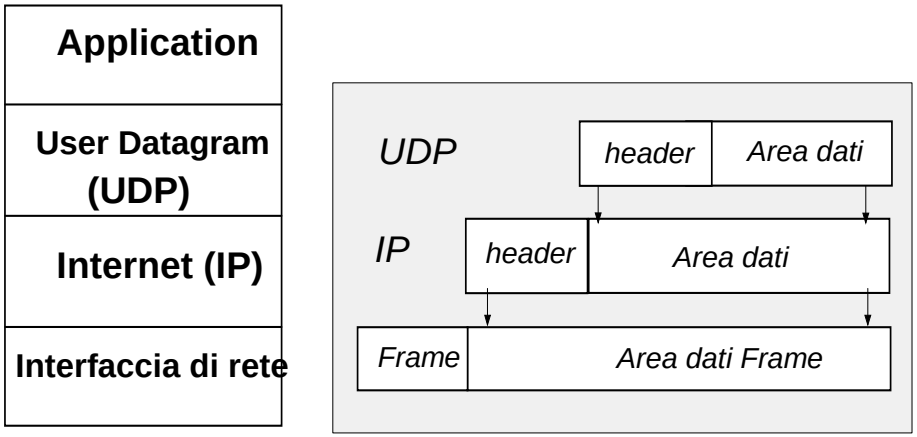
UDP User Datagram Protocol
Rete nodo a nodo
Trasporto processo a processo

- **IP** indirizzo del destinatario identifica un nodo
  IP trasferisce dati tra una coppia di nodi su Internet

- **UDP** deve distinguere tra più **processi** in esecuzione su un dato nodo connesso alla rete
  processi identificati con protocol number
  *{indirizzo: indirizzo IP + **numero di porta**}*

UDP si appoggia a IP per consegnare i datagrammi

LIVELLI CONCETTUALI



UDP fornisce un servizio unreliable e connectionless

datagrammi possono essere **persi**, **duplicati**, pesantemente **ritardati** o consegnati **fuori ordine**

il programma applicativo che usa UDP deve trattare i problemi

PROTOCOLLO

formato di un datagramma UDP

0 16 31

UDP SOURCE PORT	UDP DESTINATION PORT
UDP MESSAGE LENGTH	UDP CHECKSUM
DATA	
. . .	

I messaggi UDP sono *user datagram*

header e *area dati*

header

diviso in quattro parti di 16 bit

porta sorgente

porta destinazione

lunghezza messaggio

checksum

Uno user datagram è contenuto nell'area dati del datagramma IP

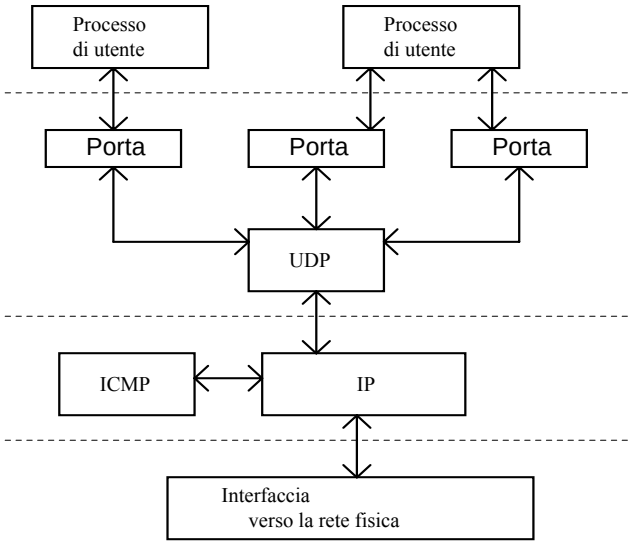
Protocollo UDP

Decisioni di

multiplexing, demultiplexing, porte

multiplexing ==> messaggi da più processi applicativi paralleli con un solo servizio IP

demultiplexing ==> lo stesso messaggio recapitato alla porta corretta



Uso di ***porte***

ogni programma ha almeno una porta per inviare/ricevere datagrammi

Lo spazio della porta UDP è descritto con nomi a 16 bit

ASSEGNAZIONE DEI NUMERI DI PORTA UDP
Autorità Centrale vs. Collegamenti Dinamici

NOMI STATICI
Autorità Centrale
per assegnare i numeri di porta universalmente validi

well-known port

0	*Riservato*
7	echo
9	discard
11	users
13	daytime
37	time
69	tfpt (trivial file transfer protocol)
111	Sun RPC protocol
513	who (demone di rwho)
514	system log

NOMI DINAMICI
Collegamenti Dinamici
assegnamento su necessità
numeri di porta non a priori, ma dati su richiesta

Realizzazione Internet UDP/TCP ==> soluzione ibrida
Alcuni numeri di porta a priori
Altri assegnati dinamicamente

SPAZI delle PORTE

Sia UDP sia TCP utilizzano il protocollo IP
ma anche svincolati da IP

connessione end-to-end
TCP comunicazione simultanea di più processi della stessa macchina

TCP crea l'astrazione di ***connessione***
coppia di estremi (*endpoint*)
Un **endpoint** è definito dalla **coppia di interi {*host,port*}**
con *host* è l'indirizzo IP dell'host della porta TCP *port*

***connessione* {*host1, port1, host2, port2*}**
un port number può essere condiviso da più connessioni

i numeri di porta non sono esclusivi ==>
servizi concorrenti

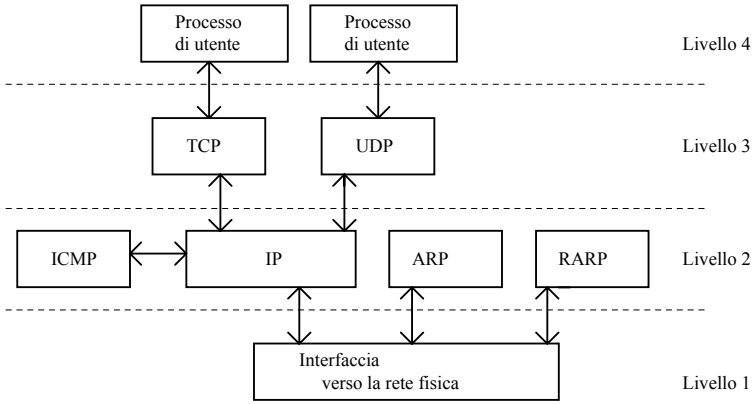
Connessioni distinte
connessione {host1, port1, host2, port2}
connessione {host1, port1, host2, port3}

Principali servizi assegnati di autorità dal NIC (Network Information Center) ad es. *posta elettronica*
numeri di porta *well-known*

Le porte TCP sono distinte dalle porte UDP

PORTA	PROTOCOLLO	DESCRIZIONE
20	FTP-DATA	File Transfer Protocol (dati)
21	FTP	File Transfer Protocol
23	TELNET	Terminale remoto
25	SMTP	Protocollo di posta elettronica
80	HTTP	Protocollo WWW
119	NNTP	Protocollo di invio news

Quadro completo delle associazioni tra porte e servizi in */etc/services*



Livelli dei protocolli di TCP/IP

TCP (Transmission Control Protocol)

Fornisce un servizio di **trasmissione dati affidabile**
basato sulle proprietà

- **reliable stream full duplex**
- **connessione o canale virtuale bidirezionale**
  la connessione end-to-end garantisce che il messaggio passa dalla memoria del mittente al destinatario con successo
- **flusso di dati non strutturato** (**byte stream**)
- **presenza di dati prioritari**
  BANDA PER DATI **NORMALI**
  BANDA LIMITATA PER DATI **URGENTI**

NON SI IMPEGNANO I NODI INTERMEDI
si usano solo le risorse degli end-user

PROTOCOLLO

- formato dei dati trasmessi (*segmenti*)
- possibilità di *dati urgenti*
- regole per la *bufferizzazione* e l'invio degli *acknowledgement* (sliding window) e relativo formato
- possibilità di *comporre* messaggi e *decomporre*
- *meccanismi di de/multiplexing* (vedi UDP)
  concetto di porta per distinguere più processi su uno stesso host

SERVIZI

- stabilire la connessione /chiudere
- scambiare dati sulla connessione

Formato del segmento TCP (header 20 byte)

0 4 10 16 24 31

SOURCE PORT			**DESTINATION PORT**	
SEQUENCE NUMBER				
ACKNOWLEDGEMENT NUMBER				
HLEN	**RSRVD**	**CODE BIT**	**WINDOW**	
CHECKSUM			**URGENT POINTER**	
OPTIONS (IF ANY)				**PADDING**
DATA				
. . .				

lunghezza del segmento in checksum

CODE BIT

URG	un dato urgente nel segmento
ACK	acknowledgement nel segmento
PUSH	invio immediato del segmento
RST	reset di una connessione
SYN	si stabilisce la connessione
FIN	termine della connessione

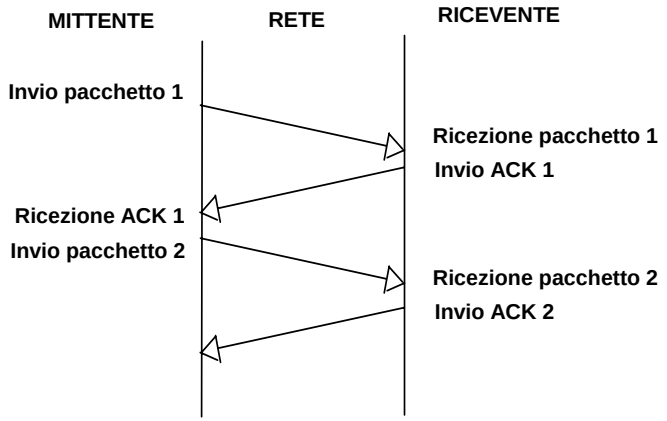
Si cerca di frammentare meno possibile i messaggi detti **segmenti** dal protocollo:

- segmenti *troppo corti*: grosso overhead di trasmissione
- segmenti *troppo lunghi*: frammentazione a livello di IP e possibili perdite ed overhead

Dimensionamenti

Reliability

richiederebbe una attesa sincrona di un messaggio di conferma (*acknowledgement* o ACK) per ogni segmento spedito prima di inviarne uno successivo
vedi **ARQ**



positive acknowledgement in ARQ

Il mittente deve attendere tra una trasmissione e l'altra
==> ***inefficienza*** del sistema di conferme
Se un ack non arriva, ritrasmissione ==>

Efficienza

Se c'è traffico nei due sensi, gli ack sono inseriti sul traffico in direzione opposta (**piggybacking**)

da cui **canale bidirezionale**

POSSIBILITÀ finestra scorrevole

finestra ideale con **dimensione in byte**

il mittente invia segmenti fino a saturare la finestra
anche senza nessuna conferma di ricezione
il destinatario invia conferme alla ricezione di un segmento

- se i segmenti spediti sono **confermati**, la finestra *scorre* e si trasferiscono segmenti successivi
- gli **ack** possono arrivare non nell'ordine di trasmissione
- se scade il **timeout** di un segmento, si reinvia

TCP usa GO BACK-N

ossia in caso *di non ricezione di un segmento*

- il ricevente può scartare quelli successivi e attendere il segmento mancante
- il mittente deve rimandare da quello che manca

reinvio fino ad una eccezione

- il ricevente deve favorire il reinvio del segmento mancante

Parametri

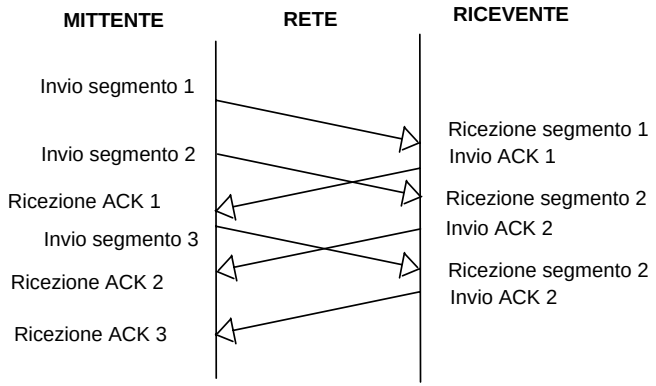
DOPO quanto tempo si ritrasmette?
QUANTE VOLTE si esegue le ritrasmissione?

Il protocollo è a stream

ossia un segmento può essere rimandato con dimensioni diverse e non ci sono garanzie di lunghezze predefinite o stabili

livelli di prestazione ==>

dimensione della finestra (stabilita dinamicamente) e **velocità** di trasmissione dei pacchetti



Tre segmenti con finestra scorrevole

Rispetto ad altri casi, TCP

- usa **byte** per dimensione della finestra
  i segmenti vengono inseriti nel flusso
- lavora con finestra di dimensione variabile
  sliding window *specificata del ricevente*
- intende gli ack in modo **cumulativo**
  un **ack** *specificato del ricevente* porta l'indicazione, di tutto ciò che è stato ricevuto nello stream fino al momento dell'ack
  in caso di perdita, si continua a mandare ack per l'ultimo ricevuto
- ritarda i messaggi che vengono inviati raggruppati in un segmento locale prima dell'invio (anche gli ack)
- usa **piggybacking** per gli ack

ack cumulativi in TCP

Arrivo di ack di un messaggio implica che sono arrivati anche i precedenti

perdita di ack non forza ritrasmissione

svantaggio

un **ack cumulativo** dice poco sullo stato del ricevente

al mittente ack dice la stessa posizione nello stream ricevente (anche se alcuni dopo arrivati)

con modo ack selettivo (vedi OSI)

- aspettare l'ack richiesto dopo la trasmissione e reinviare solo quello mancante
- reinviare tutto, anche quelli già ricevuti dal ricevente

Efficienza

evitare l'invio di segmenti corti

un messaggio corto rappresenta un forte overhead

*TCP tende a non mandare **messaggi corti***
sia per il mittente, sia per il destinatario (efficienza)

messaggi **raggruppati** al mittente

gli **ack** sono ritardati in attesa di traffico in verso opposto

peggioramento del **tempo di risposta**

specie in caso di interattività

definizione di un **time-out** oltre il quale il messaggio corto viene inviato

protocollo per stabilire la CONNESSIONE TCP

connessione tra due nodi

three-way handshake

tre fasi di comunicazione per il coordinamento

PRIMA FASE

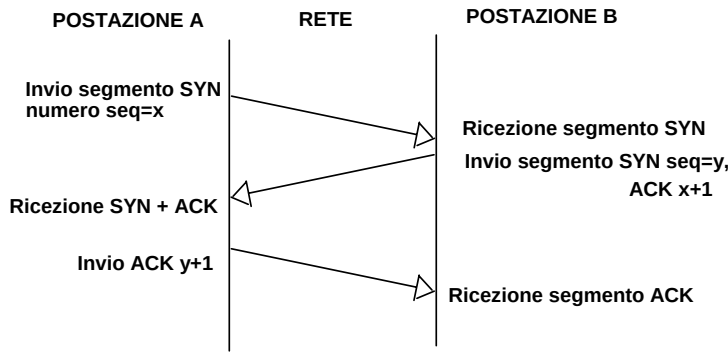
A invia il segmento SYN a B e richiede la connessione (SYN nell'header del segmento e X valore scelto da A)

SECONDA FASE

B riceve il segmento SYN e ne invia uno identico ad A insieme all'ACK (e Y valore scelto da B)

TERZA FASE

A riceve il segmento SYN ed ACK e conferma la ricezione a B attraverso un ack a sua volta



Il sistema di comunicazione a tre fasi ==> ***compromesso***
ogni nodo invia un messaggio ed ha conferma
Semantica at-most once

protocollo di BIDDING (senza rifiuto)

NEGOZIAZIONE a tre fasi per stabilire proprietà
si verifica che

- entrambi i nodi disponibili alla connessione per una sessione di comunicazione
- accordo sulla sequenza iniziale di valori: ogni pari propone per il proprio verso:
  numeri di **porta**
  numeri per i **flussi** (messaggi ed ack)
  tempo di **trasmissione e risposta** (finestra, ...)
  La sequenza é confermata proprio durante la inizializzazione
  Scelta casuale di un numero da cui iniziare la numerazione e comunicato all'altra per ogni flusso

E se si perde un messaggio?
Si attua un **time-out** con intervalli crescenti
normalmente 5,8 sec, poi 24 sec.

In fase iniziale si negoziano anche altre opzioni:

- accordo sul **MSS** (maximum segment size)
  dimensione del blocco di dati massimo da inviare
  default 536
  Maggiore il valore, migliori le performance
- **fattore di scala** della finestra
- richiesta di **tempo** e **risposta**
  per il coordinamento degli orologi

Sono possibili azioni simultanee
di apertura/chiusura

CHIUSURA

NEGOZIAZIONE
chiusura a fasi
== > semplice operazione di *close* ***graceful***
Chiusura monodirezionale
Chiusura definitiva in un verso
senza perdere i messaggi in trasferimento

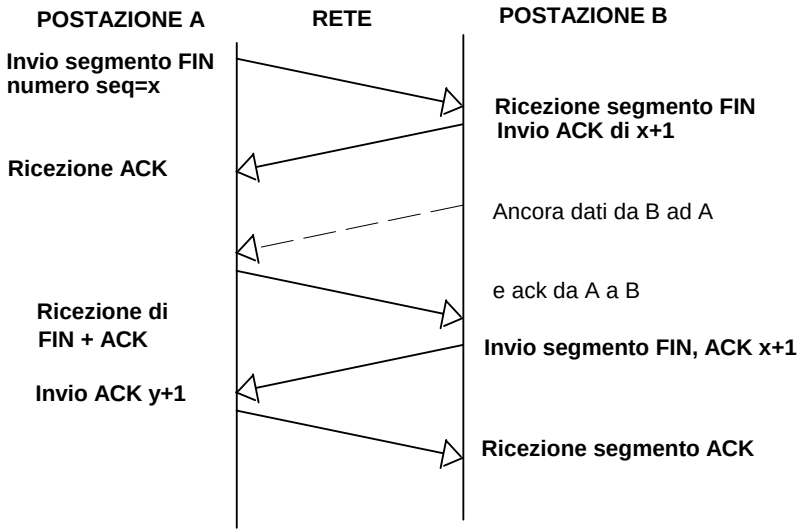
A comunica a TCP di non avere ulteriori dati e chiude
TCP chiude la comunicazione solo nel verso da A a B
se B non ha terminato, i dati continuano a fluire da B ad A
I dati che precedono la fine sono ricevuti prima della fine della connessione da A a B.

controllo ancora aperto da A a B (flusso di ack)
TCP permette solo il passaggio di ack su canale intenzionalmente chiuso

chiusura a quattro fasi

A invia segmento FIN (che arriva dopo i relativi dati)
TCP aspetta a dare corso alla chiusura definitiva, ma invia ad A solo un ack
Dopo il tempo necessario per i programmi applicativi
B invia ad A il suo segmento FIN che informa della disponibilità a chiudere la connessione

L'ultimo passo ==> **conferma da A a B** della ricezione del segmento FIN e la chiusura del collegamento



Anche per la chisura sono possibili azioni simultanee

gestione eventi anomali

segmento di ***reset*** viene inviato
per rilevare una situazione anomala
ad esempio, richiesta di connessione senza server

anche un **reset** della connessione stabilita
per abortire la stessa (i dati sono persi)

tipicamente dopo molti tentativi a vuoto
si attua un reset per tentare un ripristino

La connessione esiste solo negli endpoint
in caso di guasto a fronte di azioni ripetute di recovery
si stabilisce di chiudere in modo abortivo e unilaterale

Protocollo

situazione iniziale

three-way handshaking
in cui si stabiliscono una serie di parametri operativi per la connessione e che preparano l'avvio

situazione a regime

inizia l'uso della connessione

si comincia a lavorare..
ma si può pensare di essere subito a regime?

transitorio iniziale
regime

varie condizioni operative diverse

si devono considerare situazioni di congestione individuando
o prevenendo i colli di bottiglia
fino a ristabilire una situazione normale
o fino ad un abort della connessione

situazione finale

chiusura manifestata da uno dei due pari e accettata dall'altro
operatività con canale monodirezionale di dati, ma con messaggi di controllo in entrambe le direzioni

Protocollo a regime

Calcolo del time-out

dopo il calcolo iniziale da parte di ognuno dei due

per ogni segmento, in base al tempo di percorrenza medio **RTT andata e ritorno**

Aggiornamento del fattore in base ai valori correnti

il calcolo tiene conto sia della storia pregressa, sia del valore ricavato correntemente con pesi diversi

per non essere nè troppo reattivo e nè troppo conservativo

spesso il time-out viene calcolato come

multiplo di 100, 200 0 500 ms

Problemi in caso di calcolo su ritrasmissioni

un ack in arrivo è per il primo o per un messaggio ritrasmesso

Si sono diffusi diversi algoritmi di calcolo del time-out (Karn)

sempre tenendo conto di criteri di minima intrusione e del requisito di efficienza

Ricalcolo del Timeout

Intervallo Stimato=

α * TempoMedio + β * TempoCorrente

Controllo del flusso

controllo di flusso fondamentale per internet in cui sono presenti macchine molto diverse fra loro

La **finestra** è il meccanismo fondamentale

a cui si aggiunge

la **dimensione** preferenziale dei segmenti da inviare

attesa di dati prima di inviarli fino ad avere un segmento che sia conveniente inviare

(**M**aximum **S**egment **S**ize una delle opzioni di TCP)

Naturalmente in caso non ci sia più traffico, dopo qualche tempo si invia il segmento di qualunque dimensione sia

questo vale sia per il traffico di dati

sia per il traffico di controllo (ACK)

Possibilità Utente

in caso di segmento con **PUSH**, il segmento inviato immediatamente

in caso di informazioni urgenti (bit **URG**) se ne segnala la posizione nel flusso

in genere, si deve

Evitare di avere trasmissioni di messaggi corti

Silly window finestre limitate e messaggi brevi

in genere non si fanno azioni sotto una certa soglia

Alcune proprietà protocollo TCP

Applicazione rlogin
ogni informazione (1 char) richiede 41 byte

Una scelta praticata per evitare messaggi corti inviati indiscriminatamente
☺ **Algoritmo di Nagle** si ammette di avere pendente senza ack al più un solo messaggio corto
retroazione automatica

Applicazioni come Xwindow
disabilitano l'algoritmo di Nagle per ottenere una migliore interattività

la connessione TCP non comporta alcun uso di risorse se non si inviano messaggi
Timer
per garantire l'operatività invio di un messaggio di **keep-alive** inviato ogni intervallo (7200) sec.

A regime ogni **segmento inviato** produce **coordinamento** attuato usando l'header
si inviano sempre informazioni di controllo al pari
Ogni messaggio con **ack** specifica il flusso ricevuto e la **finestra** di accettazione corrente nella propria direzione
Il ricevente adegua la dimensione della **sliding window (di cui è mittente**)
inoltre si misurano e riadeguano i time-out

congestione
Esaminiamo i casi di congestione

Identificazione della congestione
time out che scatta in modo ripetuto:
si assume che il pari non sia raggiungibile (in congestione)

Azione locale per evitare di aggravare:
in caso di congestione *si dimezza la finestra e si raddoppia il time-out:* al termine della congestione si riparte con finestra piccola (slow start)

Evitare la potenziale congestione iniziale
se mandassimo subito tutto il flusso dichiarato dal ricevente, probabilmente causeremmo dei transitori di congestione su router intermedi, che devono scaldarsi
Le variazioni vengono fatte in modo dolce (slow start)

Si usano più **indicatori**
rwnd finestra di invio dettata dal ricevente
cwnd finestra congestione
ssthreshold soglia di slow start

La **rwnd** è considerata a regime, ma ci si arriva con un inizio che parte da finestre molto limitate che crescono in base all'assorbimento della rete e agli ACK ricevuti
con crescite differenziate
veloci inizialmente e più limitate successivamente

Controllo congestione TCP

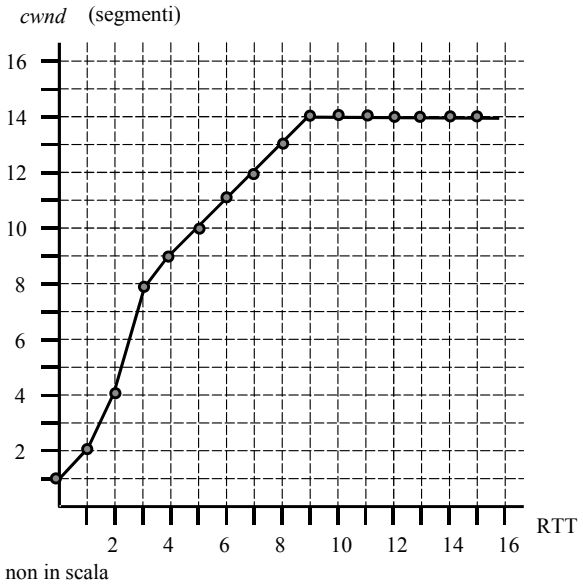
timeout come indicatore di congestione

cwnd window congestione (numero segmenti)
finestra di congestione del sender

rwnd finestra controllo flusso ricevente

ssthresh slow start threshold
soglia di base in memoria (64K)

- **slow start** all'inizio
  ***cwnd* = 1** segmento, *ssthresh* = 64K
- per ogni ACK:
  if ***cwnd*** $\leq$ ***ssthresh*** raddoppia ***cwnd***
  else ***wnd = cwnd* + 1**
- in caso di congestione
  ssthresh = *cwnd*/2
- in caso di timeout *cwnd* = 1



Scenario di uso

solo dopo la fase iniziale in cui si scambiano informazioni
numero di sequenza iniziale, finestra consentita, massima dimensione del segmento da scambiare (**Maximum Segment Size**), si calcolano time-out
si possono scambiare dati ...

in genere, **il protocollo tiene conto** di:

slow start cioè con un segmento nella *finestra di congestione*, che viene incrementata appena arriva un ack
quando la *finestra di congestione* raggiunge quella *di ricezione*, siamo a regime

silly window per evitare di lavorare un byte alla volta, non si annunciano finestre di dimensione troppo piccole (MSS/2)

ricalcolo del time-out in modo dinamico
il time out corrente viene tarato rispetto a quanto calcolato come media per la stima del nuovo

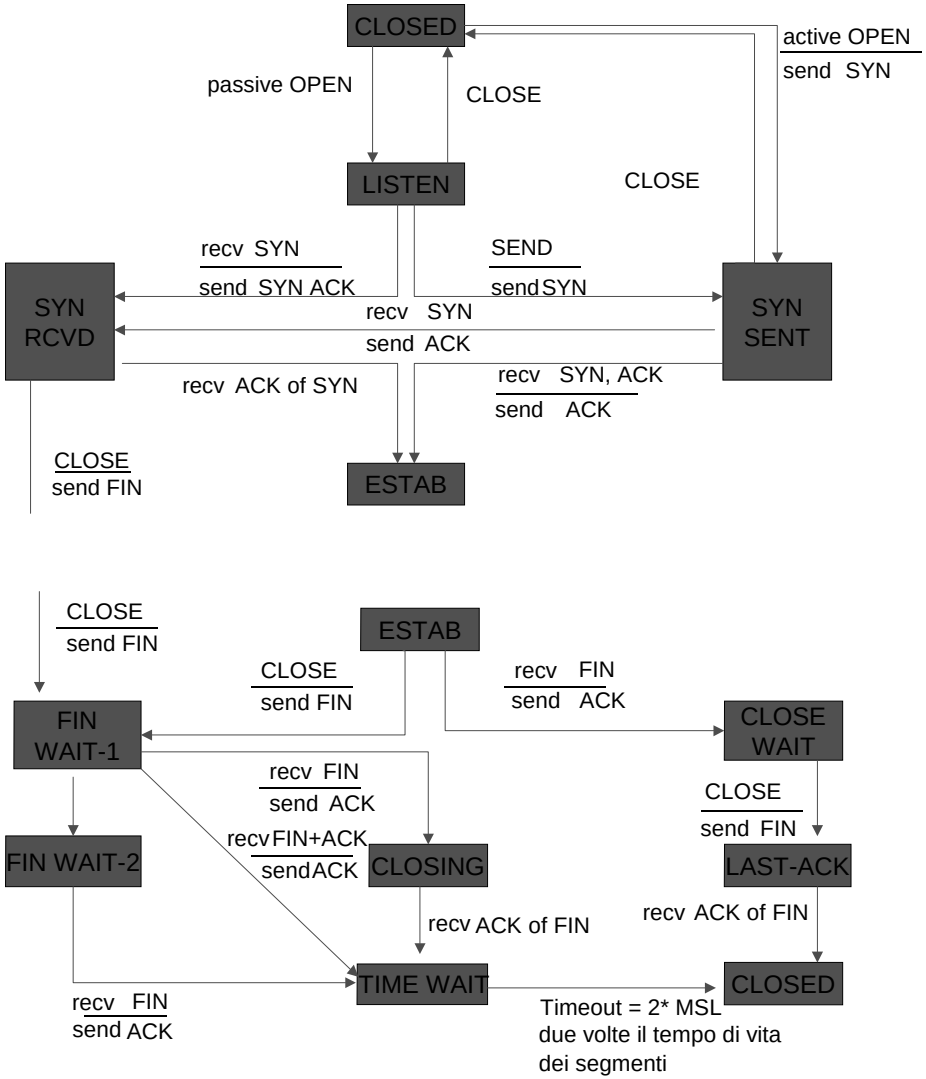
exponential backoff in caso di ritrasmissione, il timeout raddoppia, dopo raddoppia ancora, fino ad un massimo
(ad es. 4') poi si chiude la connessione

long fat pipes mantenere piene pipe a banda elevata (fornendo indicazioni di buffer superiori a quelli di utente
e bufferizzando a livello di supporto)

limiti al time-wait la memoria su una porta viene mantenuta per tempi sempre inferiori

Diagramma degli stati di TCP

sono stati proposti spesso diagrammi degli stati per modellare lo stabilirsi della connessione e per la chiusura



Problemi di TCP

In caso di un uso estensivo di strumenti applicativi che seguano un protocollo:

cliente **apre connessione**
richiesta 1 messaggio
chiusura connessione

servitore **attesa connessione**
ricezione 1 messaggio/risposta
chiusura connessione

le azioni di prologo/epilogo dominano
(si vedano le interazioni Web)

ESTENSIONI a TCP

è stato proposto il **T/TCP** che raggruppa le azioni:

- il cliente manda il messaggio già con il primo segmento e si rinuncia alla fase di negoziazione
- il servitore all'arrivo del messaggio invia con l'ack, la risposta e la chiusura della connessione

☺ nel caso migliore (se non ci sono problemi), il protocollo costa come UDP
(1 messaggio andata, 1 di ritorno)

☹ nel caso peggiore, il costo si avvicina a quello di TCP

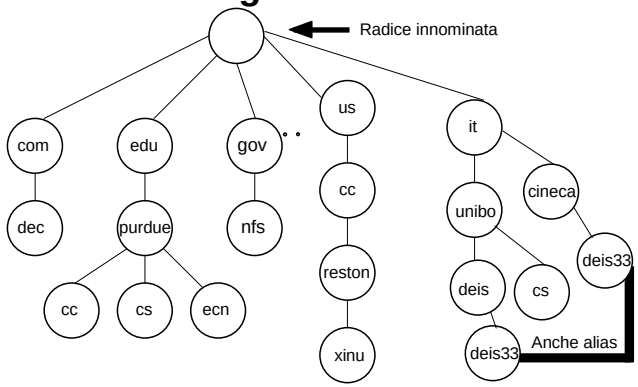
DOMAIN NAME SYSTEM (DNS)

Insieme di gestori di tabella di nomi logici e di indirizzi IP
obiettivo principale => attuare corrispondenze
tra nomi di host e indirizzi IP

Primo passo: /etc/hosts
Non sufficiente

NOMI LOGICI GERARCHICI

Gerarchia di **domini logici**



la corrispondenza tra **nomi logici** e **indirizzi fisici** avviene dinamicamente tramite un servizio di nomi che risponde (**dinamicamente**) alle richieste di traslazione

La traslazione:
statica vs. dinamica
locale vs. globale
non una gestione **globale centralizzata** o **statica**

Esempio di ***divisione dei compiti e coordinamento***
replicazione partizionamento

NOMI di DNS gerarchici

Ogni nome rappresenta un dominio e può identificare sia un host sia un ulteriore insieme di nodi

Nome dominio	Significato
COM	Organizzazioni commerciali
EDU	Istituzioni per l'istruzionre
GOV	Istituzioni governative
MIL	Gruppi militari
NET	Maggiori centri di supporto alla rete
ORG	Organizzazioni diverse dalle precedenti
ARPA	Dominio temporaneo dell'ARPANET (obsoleto)
INT	Organizzazioni internazionali (schema geografico)
codice nazionale	Ciascuna nazione (schema geografico)

deis33.cineca.it a tre livelli
NOME con vari identificatori (o *label*) ciascuna un dominio

Livello	Descrizione	Nome dominio	Sigle
minimo	locale	deis33.cineca.it	deis33 = macchina
intermedio	gruppo	cineca.it	cineca = gruppo
massimo	organizzazione	it	it = Italia

deis33.deis.unibo.it

country	it = Italia,
organisation	unibo = Università di Bologna,
dept	deis = Nome/Sigla Organizzazione locale,
machine	deis33 = nome della macchina,

Livello	Descrizione	Nome dominio	Sigle
minimo	locale	deis33.deis.unibo.it	deis33 = Host
intermedio2	sottogruppo	deis.unibo.it	deis = Organisation
intermedio1	gruppo	unibo.it	unibo = U of Bologna
massimo	postazione	it	it = Italy

Nomi di DNS

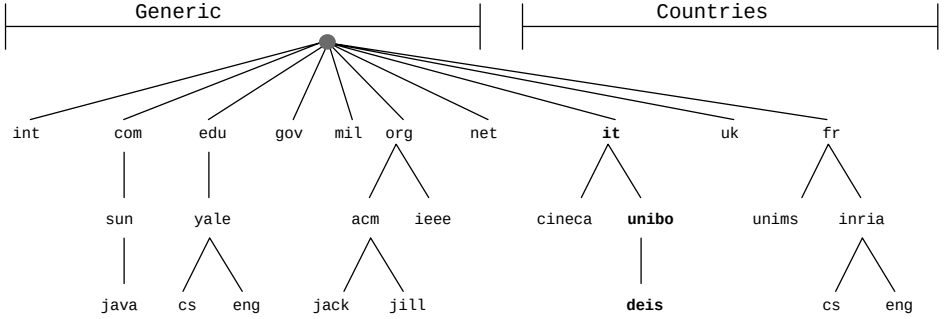
I singoli nomi sono **case insensitive** e al max ***63 char***
Il nome completo al max 255 char

I domini non sono collegati in nessun modo alle reti fisiche o alle organizzazioni (***logico vs. fisico***)

Ogni dominio indicato in modo **relativo** o **assoluto**

Ogni dominio deve fare riferimento al dominio che lo contiene **deis.unibo.it**
deis è interno a **unibo**, che è interno a **it**, che è interno alla root

Possibile gerarchia



Concetto di delega

un dominio delega i sottodomini a gestori sottostanti
(che se ne assumono responsabilità e autorità)

Implementazione DNS

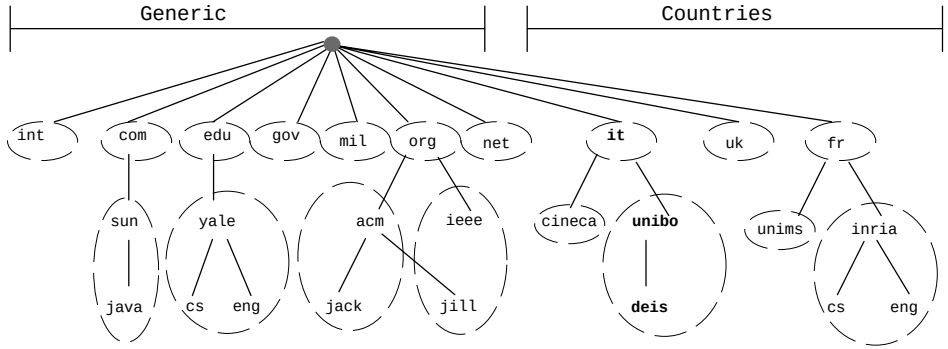
Ogni richiesta viene fatta al servizio di nomi tramite un **agente specifico** di gestione dei nomi per una località

a livello di API si passa il riferimento da mappare ad un **resolver** che
- o conosce già la corrispondenza (cache)
- o la trova attraverso una richiesta C/S a un **name server**

I Domini sono divisi in **zone** di autorità soggette a diversi **servitori**
che possono delegare anche altri della gestione

Diversi requisiti => **affidabilità**, **efficienza**, **località**

suddivisione in **zone**
geografica o di organizzazione



Ogni **zona** riconosce una autorità che fornisce le corrette corrispondenze

Diversi DNS come domini separati

Ogni dominio corrisponde al **Name Server** che ha autorità sulla **traslazione degli indirizzi**
che non ha una visione completa, ma solo locale

In genere, ogni zona ha un **primary master**
responsabile per i dati della intera zona

ma in più ci sono una serie di **secondary master**
che sono copie del primary, con consistenza garantita dal protocollo DNS (non massima)

Reliability

allo start up il secondario chiede i dati al primario e può fare fronte al traffico in caso di guasto
Ad intervalli prefissati, i secondari chiedono le informazioni al primario (modello pull)

È bene avere più server master per zona
I **ruoli** sono mescolati in modo libero: primario di una zona può diventare il backup (master secondario) di un'altra zona

Efficienza su località

i dati ottenuti possono essere richiesti nuovamente
i server mantengono informazioni
caching *dei diversi server per ottimizzare i* ***tempi di risposta*** al cliente

Protocollo di **richiesta** e **risposta** per il **name server**
con uso di protocollo **UDP** (comunicazione porte 53)
e se messaggi troppo lunghi? Eccezione e uso di **TCP**

DNS

Un server mantiene un record per ogni risorsa **dinamico**
(caricato da file di configurazione ed aggiornato)
Le query consultano l'insieme dei record

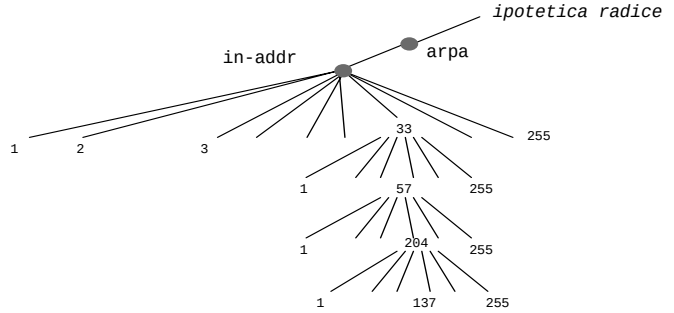
- **Nome dominio**
- **Time to live** (tempo validità in secondi)
- **Classe** (**Internet IN**)
- **Tipo** (descrizione del tipo)
- **Valore**

I tipi significativi

Tipo	Significato	Valore
SOA	Start of Authority	parametri della zona
A	IP host address	intero a 32 bit (dot not.)
MX	Mail exchange	server di domino di mail
NS	Name server	server per dominio corrente
CNAME	Canonical name	alias di nome in un dominio
PTR	Pointer	per corrispondenza inversa
HINFO	Host description	descrizione di host e SO
TXT	Text	testo qualunque

Sono possibili anche accessi e query inverse: ossia PTR
si entra con l'**indirizzo** e si ottiene il **nome**

Il tutto richiede (?) un record per ogni corrispondenza inversa...



Esempio di record DNS

```
@   IN SOA promet1.deis.unibo.it.  postmaster.deis.unibo.it.
    (644 10800 1800  604800  86400)
; serial number, refresh, retry, expiration, TTL in sec
; versione          , 3 ore     , 1/2 o, 1 sett.       ,  1 d
  IN NS   promet1.deis.unibo.it.
  IN NS   promet4.deis.unibo.it.
  IN NS   almadns.unibo.it.
  IN NS   admii.arl.army.mil.

localhost   IN  A    127.0.0.1
@   A       137.204.59.1
  MX 10  deis.unibo.it.
  MX 40  mail.ing.unibo.it.

lab2    IN NS lab2fw.deis.unibo.it.
lab2fw IN A 137.204.56.136
amce11    IN     A     137.204.57.244
  IN HINFO       HW:PC IBM       SW:WINDOWS 95
  IN WKS         137.204.57.244  TCP FTP TELNET SMTP
  IN MX 40       amce11.deis.unibo.it.
labvisione   IN CNAME    csite27
deis18 IN TXT "Qualunque testo non significativo"
deis18 IN RP root.deis.unibo.it luca\.ghedini.mail.ing.unibo.it
; record per responsabile

@   IN SOA promet1.deis.unibo.it. postmaster.deis.unibo.it.
    (644 10800 1800   604800  86400)
  IN NS   promet1.deis.unibo.it.
  IN NS   promet4.deis.unibo.it.
  IN NS   almadns.unibo.it.
  IN NS   admii.arl.army.mil.
146     IN   PTR       deiscorradi.deis.unibo.it.
; record per la corrispondenza inversa
```


DNS Risoluzione nomi

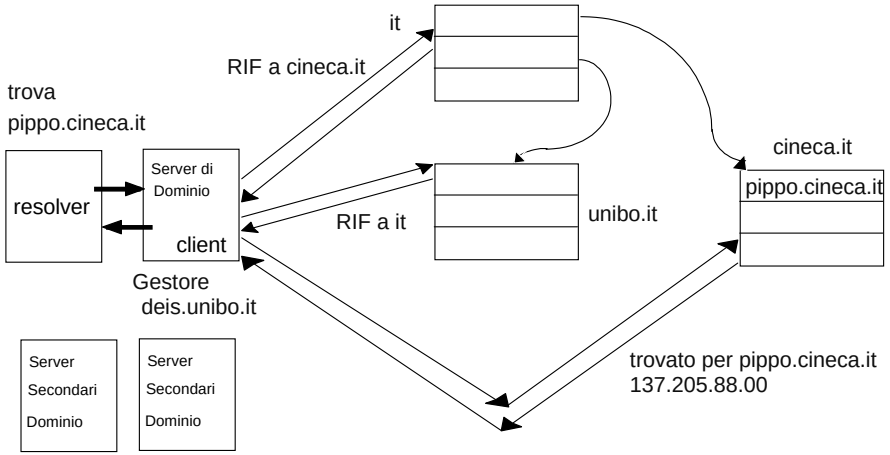
Alcune scelte dipendenti dalla implementazione
il **resolver** conosce un server di dominio
e attua le **query**

Il protocollo DNS regola gli scambi di informazioni tra server:
due tipi di query **ricorsiva** e **iterativa**

La **ricorsiva** richiede che al cliente
- si fornisca risposta (anche chiedendo ad altri)
- si segnali errore (dominio non esistente, etc.)

La **iterativa** richiede che al cliente si fornisca
- la risposta
- o il migliore suggerimento
  come un riferimento al migliore name server

Il resolver query **tipicamente ricorsiva**
il **server** di dominio si incarica di rispondere
coordinandosi con altri (**query iterativa**)



Risoluzione dei nomi

Il servizio distribuito e a **dati partizionati** ottenuto scambiando *query* tra **DNS server**

Se il server possiede il **nome**, risponde

Se query ricorsiva, cerca altre risposte e rimane impegnato fino a **risposta** o **time-out**

Se query iterativa, il server che non possiede il nome risponde con un riferimento al **gestore superiore** più vicino che possa rispondere
(si continua a scalare la gerarchia in modo dinamico, senza conoscerla a priori)

Il name server locale fa **query iterativa**, senza conoscere a priori la gerarchia (località)

Ogni name server decide se e come rispondere
Il name server root **non** accetta **query ricorsive**
anche altri che devono fornire sempre servizi

Si usano **timeout** ed eventualmente il server ne consulta successivamente altri e se le zone hanno secondari, ci si rivolge a quelli

Tipicamente, si provano diversi server noti mandando a ciascuno almeno due o più ritrasmissioni con time-out crescenti (**4** volte)
se non c'è risposta si assume che il server sia fallito
(***timeout*** = ***server crash***)

BIND (**B**erkeley **I**nternet **N**ame **D**omain)
Implementazione di Berkeley di DNS
La prima significativa come ampiezza e poi diffusione

comando ***nslookup*** come *ambiente di interrogazione* per le *corrispondenze e le inverse*
comando ***nstest*** come *shell di interrogazione*
I resolver sono invocati anche all'interno di applicazioni
rlogin, telnet ...

Il server è il demone named in etc
`/etc/named` lanciato al boot

Ogni risorsa è registrata in un **RR** - **R**esource **R**ecord
di classi diverse e mantenuta dai nodi server

I database sono ottenuti da file di **configurazione** con nomi fissati o indicati in file di nome fisso
db.137.204.57
db.DOMAIN
db.cache
db.127.0.0

Molti particolari implementativi

In genere, i **server** sono **paralleli**
i trasferimenti di informazioni sono fatte da processi figli, che si coordinano con il genitore

si usano **cache di risultati negativi**
si mettono in cache anche risposte di errore o eccezione (dominio non esistente)

Server

I server primari e secondari possono appartenere a domini diversi
la **non località** può portare a operazioni di aggiornamento costose

Si possono avere dei gestori secondari partizionati che mantengono un sottodominio ed una località

partial-secondary server

☺ se informazioni e richieste locali, molto vantaggiosi

Si possono avere dei puri gestori di record che sono solo capaci di mantenere entry cached e rispondere a query inverse

caching-only server

☺ che sono cache per i nodi del dominio

server interni a una località che accettano query ricorsive per fornire sempre risposta

forwarder

☺ che alleggeriscono il peso nel dominio

Si cominciano a considerare caratteristiche di **sicurezza**

- limitando i clienti che possono accedere (e le operazioni che possono richiedere)
- limitando le zone che altri name server possono chiedere

> unibo.it.

Server: promet1.deis.unibo.it Address: 137.204.59.1
res_mkquery(0, unibo.it, 1, 1)
Got answer: HEADER:
opcode = QUERY, id = 15, rcode = NOERROR
header flags: response, want recursion, recursion avail.
questions =1, answers =1, authority records = 4, additional =4

QUESTIONS: unibo.it, type = A, class = IN
ANSWERS:
-> unibo.it internet address = 137.204.1.15
ttl = 58196 (16 hours 9 mins 56 secs)
AUTHORITY RECORDS:
-> unibo.it nameserver = dns2.cineca.it
ttl = 155488 (1 day 19 hours 11 mins 28 secs)
-> unibo.it nameserver = dns.nis.garr.it
ttl = 155488 (1 day 19 hours 11 mins 28 secs)
-> unibo.it nameserver = dns.cineca.it
ttl = 155488 (1 day 19 hours 11 mins 28 secs)
-> unibo.it nameserver = almadns.unibo.it
ttl = 155488 (1 day 19 hours 11 mins 28 secs)
ADDITIONAL RECORDS:
-> dns2.cineca.it internet address = 130.186.1.1
ttl = 258410 (2 days 23 hours 46 mins 50 secs)
-> dns.nis.garr.it internet address = 193.205.245.8
ttl = 119860 (1 day 9 hours 17 mins 40 secs)
-> dns.cineca.it internet address = 130.186.1.53
ttl = 258410 (2 days 23 hours 46 mins 50 secs)
-> almadns.unibo.it internet address = 137.204.1.15
ttl = 414688 (4 days 19 hours 11 mins 28 secs)

Non-authoritative answer:
Name: unibo.it Address: 137.204.1.15

> **cineca.it.**
Server: promet1.deis.unibo.it res_mkquery(0, cineca.it, 1, 1)

Got answer: HEADER:
  opcode = QUERY, id = 28, rcode = NOERROR
  header flags: response, want recursion, recursion avail.
  questions =1, answers =0, authority records =1, additional = 0
QUESTIONS: cineca.it, type = A, class = IN
AUTHORITY RECORDS:
   -> cineca.it
      ttl = 10784 (2 hours 59 mins 44 secs)
      origin = dns.cineca.it
      mail addr = hostmaster.cineca.it
      serial = 1999052501
      refresh = 86400 (1 day)
      retry   = 7200 (2 hours)
      expire = 2592000 (30 days)
      minimum ttl = 259200 (3 days)

Name: cineca.it

; query inverse
> **set q=ptr**
> **86.57.204.137.in-addr.arpa.**
Server: promet1.deis.unibo.it Address: 137.204.59.1
86.57.204.137.in-addr.arpa name = deis86.deis.unibo.it
57.204.137.in-addr.arpa nameserver = admii.arl.army.mil
57.204.137.in-addr.arpa nameserver = almadns.unibo.it
57.204.137.in-addr.arpa nameserver = promet4.deis.unibo.it
57.204.137.in-addr.arpa nameserver = promet1.deis.unibo.it
admii.arl.army.mil internet address = 128.63.31.4
admii.arl.army.mil internet address = 128.63.5.4
almadns.unibo.it internet address = 137.204.1.15

Progetto di DNS e domini

Criteri di progetto per una organizzazione

1. stabilire il **dominio/domini** come raggruppamento logico di nodi
2. per ogni nodo, è necessario avere a disposizione il servizio, e, quindi, almeno **un servitore**
3. raggruppare i nodi per **località** in **sottoinsiemi** e determinare anche gli eventuali **sottodomini**
4. allocare i server su macchine che siano **visibili** facilmente a sottogruppi ma non troppo esposti e carichi
5. ottenere servizi **replicati** attraverso server **secondari** (che si scambiano i ruoli) e anche un server **esterno**

Esigenze e Problemi residui
- connessione facile e servizio efficiente
- uso di software robusto ed flessibile
- apertura alla eterogeneità
- sicurezza

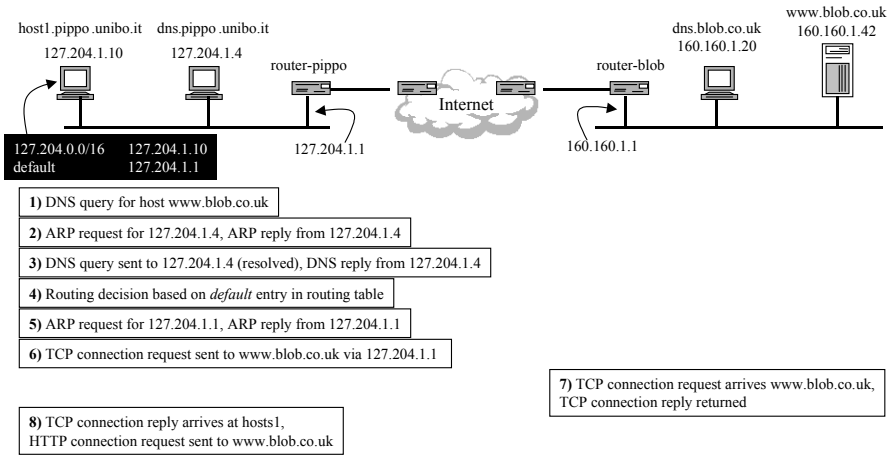
Controllo dinamico
strumenti che verificano il carico del server
anche migliaia di richieste cliente contemporanee
Tipicamente, si usano strumenti come la scrittura periodica o forzata di eventi su file di log
sia eventi di carico, sia eventi statistici, sia anomalie

Come migliorare
tenere conto del ***carico di applicazione*** (query) e
del ***carico di protocollo*** (aggiornamenti)
Adeguare l'architettura al traffico previsto

Uso dei diversi livelli di protocolli

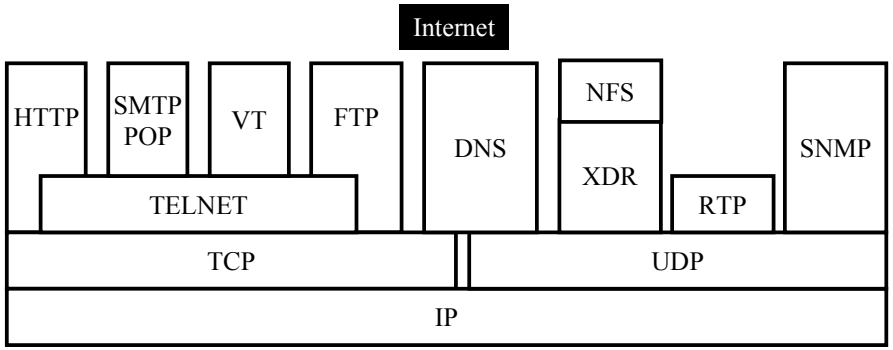
Da cliente a servitore remoto

- Richiesta di pagina Web
- Richiesta al DNS
- Protocollo ARP per trovare percorsi in uscita
- Connessione TCP tra client e server
- Passaggio dei dati
- Chiusura



Estensioni dei protocolli TCP/IP

Internet accoglie e consente lo sviluppo di molte direzioni



Accanto ai protocolli tradizionali, compaiono molte linee di sviluppo, incoraggiate da IETF con la costituzione di gruppi di lavoro

Alcuni protocolli rappresentano

necessità di ampie **utenze**: NAT, DHCP, PPP, ...

estensioni per consentire una migliore **sicurezza**

estensioni per la gestione della **mobilità**

estensioni per considerare sistemi a **flusso di informazioni multimediali**

estensioni per la gestione della **qualità di servizio**

Estensioni di servizi

Per Utenti MODEM

Uso di protocolli che partono dalla considerazione che ci sono meno utenti attivi dei potenziali utenti

☺ non è necessario avere un nome IP per ogni cliente potenziale

Pool di indirizzi

- assegnati dal pool ad ogni richiesta di un cliente (e.g. via protocolli per linea seriale **PPP** o **SLIP**)
- usati nella sessione di **modem**
- restituiti al pool al termine della sessione

sessioni diverse usano indirizzi IP diversi

I protocolli di linea seriale
servono per passare informazioni su linea punto-a-punto tra due entità (cliente e provider)

indipendentemente dall'IP o meno della macchina cliente

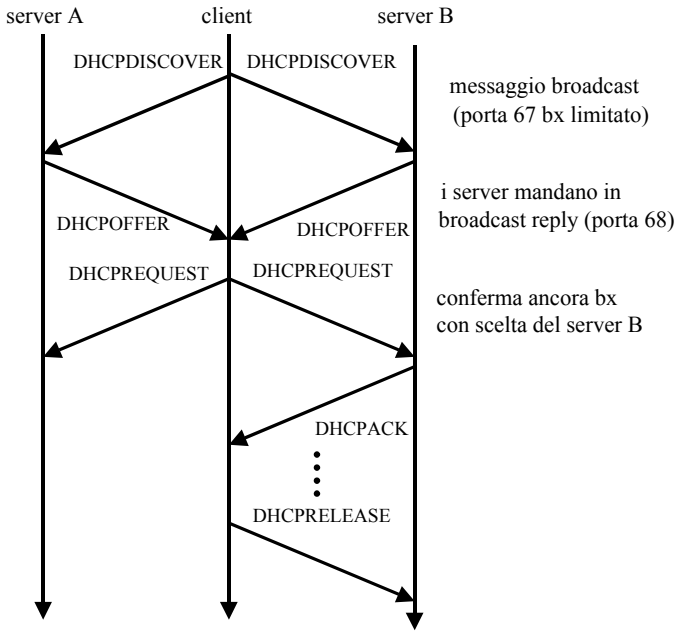
Vedi
RARP usato anche come primo passo nella inizializzazione delle macchine diskless

DHCP (rfc 2131)
Dynamic Host Configuration Protocol

Usato per la **configurazione dinamica** in IP
per host che non hanno bisogno di nomi IP permanenti
ad esempio i provider, che assegnano dinamicamente i nomi di IP assegnati loro

Si basa su due ruoli: clienti e servitori con protocollo di bidding a fasi

- broadcast del **discovery** (richiesta di ingresso)
- offerte dei servitori (con parametri di scelta)
- **scelta** di una offerta (in broadcast)
- conferma della offerta
- messaggi di **mantenimento** prima della scadenza (**lease**)



Protocollo DHCP

di molto interesse per le moderne organizzazioni
molti host di una organizzazione
(evitando set-up manuale o statica e per ragioni di sicurezza)

ovviamente manca di **sicurezza**

il **lease**, cioè si associa una durata al contratto
come un soft-state consente di riusare la attribuzione dopo un certo tempo
permette di confermare l'uso, senza rifare il protocollo

coordinamento tra server e relativi messaggi

il **DHCP** non si usa solo ma anche per la attribuzioe di una serie di parametri di gestione: maschera di rete, sottorete, diritti, ecc.ecc.

Implementazione:
☹ usa il broadcast delle LAN (protocollo locale)
☹ richiede server che memorizzano tutte le informazioni relative ai clienti gestiti

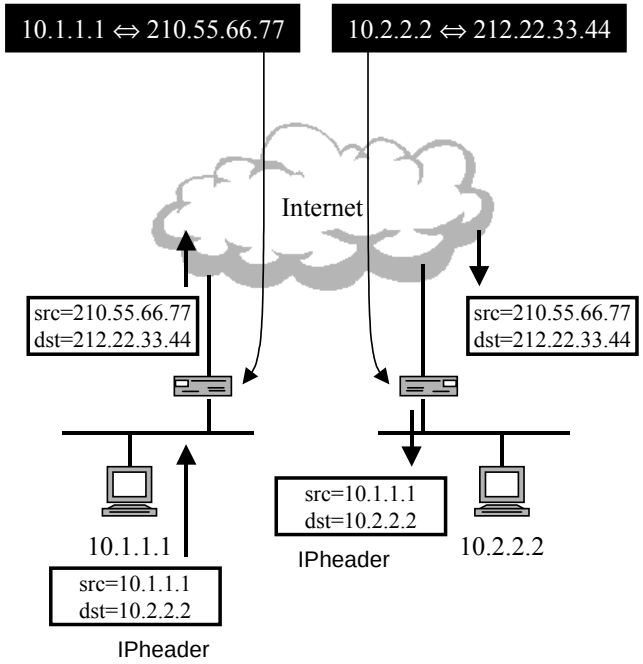
☺ molto utile ed utilizzato per
host mobili, sistemi wireless, dispositivi limitati da fare entrare in una rete di una organizzazione

NAT
Network Address Translation

Usato per **traslare indirizzi privati** in indirizzi IP globali in rete aperta (uso di indirizzi specificati in RFC1918)

- ricalcolo dei checksum per i protocolli IP, TCP, UDP

Problemi per applicazioni che usano **nomi IP** a livello applicativo



Estensioni di sicurezza

IPSEC

IPSEC è stato definito secondo alcune linee guida

I canali sono solo monodirezionali

nozione base è quella di una **security association**
che lega due entità che vogliono collaborare

Se un cliente e un servitore vogliono comunicare, si devono gestire due canali separati

SA definisce una politica unidirezionale

SPI - security parameter index
agganciato a protocollo e IP specificato
SPI permette di reperire le chiavi della associazione SA e di gestire tempi di vita della associazione SA

Il protocollo gestisce creazione, negoziazione, modifica e cancellazione della SA

Si parte con la creazione di un canale sicuro, e poi si negoziano le diverse sessioni

Si deve gestire la SA completamente, considerando

- il coordinamento di tutti gli intermedi e
- una gestione di indici associati alle operazioni per stabilira un algoritmo di gestione delle chiavi

La sicurezza introduce la **necessità di stato** sulla intera connessione

Diverse strategie

Si propongono due meccanismi

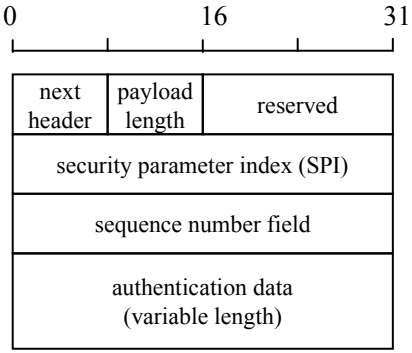
- **Authentication Header (AH)**
- **Encapsulating Security Payload (ESP)**

ciascuno con algoritmi a default
opzionali in IPv4 e obbligatori in IPv6

Authentication Header

si fornisce **integrità** e **autenticazione** basata sul singolo messaggio usando un header che autentica l'origine del datagramma (il **datagramma** in chiaro)

una **firma digitale** aggiunta nell'header del messaggio
uso di una firma con un hash a chiave **HMAC**
algoritmi MD5 o SHA



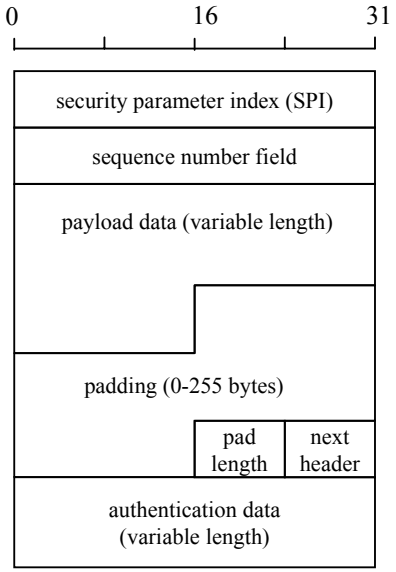
Encapsulating Security Payload

offre confidenzialità cifrando sia **dati** sia **intestazione** attraverso due modalità di lavoro

- **Trasporto**
- **Tunnel**

In modo **trasporto** il dato applicativo (livello 4) viene cifrato alla trasmissione e decifrato alla ricezione
In modo **tunnel** tutto il pacchetto corrente viene cifrato ed incapsulato in un **nuovo pacchetto** IP in chiaro

In ogni caso, il pacchetto è esteso per specificare le operazioni di incapsulamento ESP
Si usa a default il **DES** (CBC mode)



Protocollo IGMP

Internet Group Management Protocol (IGMP)
Non solo sono riconosciuti indirizzi **broadcast**
ma indirizzi multicast (in **classe D**)
distinti in **temporanei** e **permanenti**
senza una garanzia di consegna completa
È necessario avere in ogni contesto degli
agenti di multicast (un/alcuni bridge/router)

INVIO
Un processo manda il messaggio all'agente locale
RICEZIONE
Un processo che è interessato a rispondere ad un indirizzo deve registrarsi prima di essere riconosciuto
PROPAGAZIONE LOCALE
Per ogni contesto dopo le **registrazioni**, l'**agente** conosce la lista dei destinatari (se più processi sullo stesso nodo, si riceve localmente e si smista qui)
PROPAGAZIONE GLOBALE
oltre alle consegne locali, è necessario smistare agli altri contesti
(cioé ai **gateway conosciuti**)

Supporto di data link al Multicast

Traslazione **indirizzi in indirizzi MAC**
Risoluzione algoritmica
veloce, facile e distribuita
In ETHERNET: il formato indirizzi MAC
IANA suggerisce un range MAC
01:00:5e:**00:00:00 /** 01:00:5e:**7f:ff:ff**
che si mappa 224.0.0.0 - 239.0.0.0
liberi gli ultimi 23-bit della classe D
si richiede host filtering a livello IP

Estensioni per multicast

IP multicast e IGMP

IP multicast ha obiettivi diversi

il multicast consente di mandare un unico pacchetto a più destinatari, usando nomi di classe D per identificare gruppi

⇨

necessario un **supporto di rete** e di **nodi** per ottenere il coordinamento

IN GENERE, si deve costruire un albero dal **trasmettitore** come radice ai **riceventi** come foglie

con **nodi foglie** (host) e **gestori intermediari** (router)

- Il gruppo è **aperto** e anche altri possono mandare messaggi al gruppo
- l'appartenenza al gruppo è **dinamica**
- il join al gruppo è a carico delle foglie

i router sono responsabili del routing
i meccanismi di delivery sono locali

Il controllo del membership al gruppo con IGMP

IGMP prevede due soli messaggi

IGMPQUERY mandato da un router per verificare la presenza di host che rispondono ad un indirizzo

IGMPREPORT mandato dai nodi per segnalare cambiamento di stato nei confronti del gruppo

Ogni rete locale prevede almeno un router di IGMP che gestisce il traffico locale in arrivo o in partenza

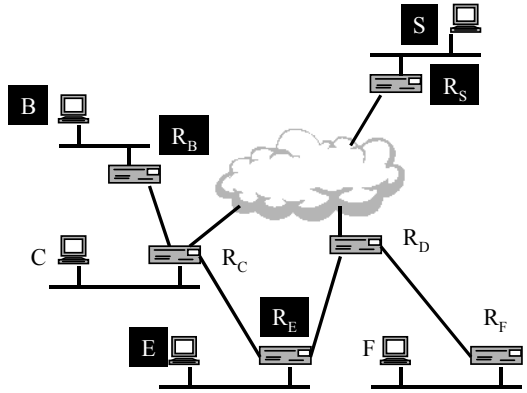
Routing Multicast

un protocollo efficiente di routing che supporti il multicast

protocolli multicast

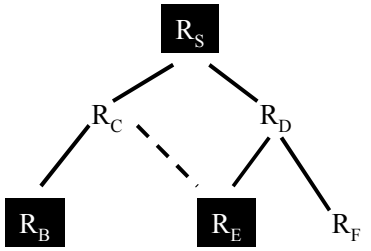
Consideriamo il caso di un host **S** che trasmette e di **B** ed **E** parte del gruppo riceventi

Si comincia mandando un messaggio **in flooding** verso tutti i possibili paretecipanti (**backbone** multicast)



Primo raffinamento

si crea un albero **spanning tree** che contiene i nodi foglia desiderati, usando le informazioni del protocollo di routing unicast



alcuni nodi sono raggiunti da più cammini

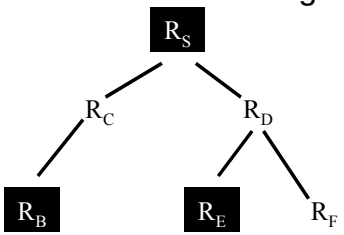
Primo raffinamento

L'albero viene identificato scartando tutti i percorsi che non servono a raggiungere le foglie presenti

reverse path broadcast (RPB)
si passa il datagramma se arriva dalle folgie alla radice con un cammino corretto e si eliminano tutti gli altri

Secondo raffinamento

eliminazione dei cammini duplicati
mantenendo informazioni di routing



Le realizzazioni devono tenere conto dai normali algortimi di routing, tipicamente:

- **Distance vector**
  deve usare informazioni next hop
  (o usare **poisoned reverse**) per bloccare
- **Link state**
  deve costruire tutti gli alberi shortest path per tutti i nodi
  si devono usare regole “tie break” per dirimere conflitti

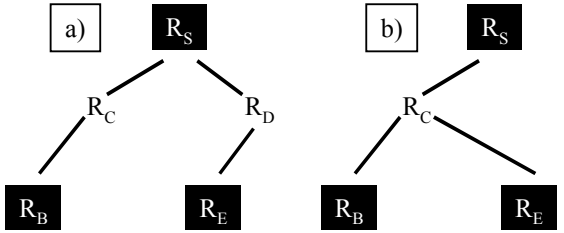
Terza raffinamento

si procede escludendo quei router che non hanno al momento alcun ricevente attraverso messaggi di taglio
pruning (potatura)

reverse path multicasting (RPM) per consentire il reinserimento di parti dell'albero (**pruning** e **graft**)

RPM

- usato in molti protocolli multicast
- mantenuto lo stato per-sender, per group



Reti che non hanno membri sono tolte dall'albero
Si deve consentire di rientrare nell'albero in caso di nuove adesioni al gruppo senza rioganizzare ex novo

graft esplicito
nodi in basso si aggiungono all'albero

soft-state (stato software)
timeout
nodi in basso fanno pruning a loro volta

MULTICAST routing

Protocolli

Molti protocolli diversi a livello di routing (incompatibili)
Al momento c'è una varietà di sforzi anche in competizione e di algoritmi supportati da comunità diverse

DVMRP (RFC 1075) di tipo RPM
basato su RIP modificato
molto usato in MBONE (multicast backbone)

MOSPF (RFC 1584)
di tipo link-state adatto per reti grandi
- RPM
- soft-state
  gli alberi sono valutati quando il pacchetto arriva

Protocol Independent Multicast PIM (RFC 2117)
può usare qualunque protocollo unicast
con due modi adatti per popolazioni diverse
- **Denso**: RPM
- **Sparso**: graft esplicito dell'albero

Core Based Trees CBT (RFC 2201)
adatto per router core
si mantengono **alberi unici** senza definire uno stato per ogni sender, e per ogni gruppo
possono risultare anche **alberi sub-ottimi**

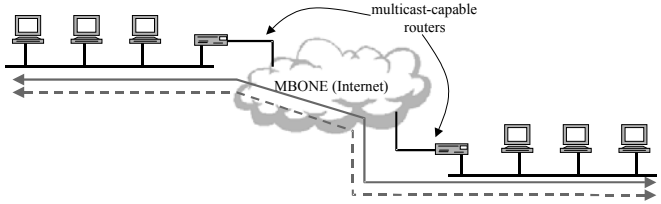
Multicast address management

Alcuni indirizzi sono riservati ma non si è manifestata la esigenza di un controllo centrale
Indirizzi generati pseudo-random
La gestione del MULTICAST diventa fondamentale per una serie di applicazioni

Estensioni per ambienti multimediali

Applicazioni multimediali integrate o meno con Internet
Applicazioni Multimedia per
- voce a audio *RAT, RealAudio*
- video *VIC*
- testo *NTE*
- whiteboard *WBD*

In generale usano tutte sessioni multicast e RTP/RTCP
Ogni sessione apre due **canali multicast** per applicazione
uso di protocolli ad hoc
porte D e due porte di utente (K, K + 1)
Nel caso si voglia rendere nota la conferenza
servizio di nomi **Session Directory RendezVous** SDR
che consente ci ottenere informazioni sul flusso



Per queste applicazioni, sono necessari **router** che siano in grado di garantire il corretto QoS
Inoltre, applicazioni diverse possono anche interagire tra di loro e richiedono di tenere conto anche di questo

Estensioni per la mobilità

MOBILE IP

il supporto per la mobilità passa attraverso due tipi riconosciuti di mobilità

user mobility ritrovare il proprio profilo da dovunque ci si colleghi (browser preference, URL recenti, etc)

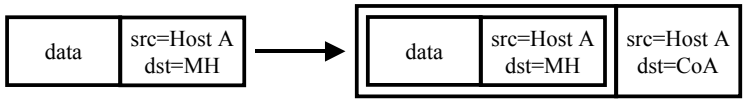
terminal mobility essere riconosciuti e ritrovati da dovunque ci si colleghi

Il supporto ad **utenti mobili** richiede trasparenza ai livelli alti senza cambiare IP per tutte le applicazioni

bisogna cambiare informazioni di routing
usando un care-of-address (CoA) per l'host mobile

- **M**obile **H**ost (**MH**)
- **H**ome **N**etwork (HN), **H**ome **A**gent (HA)
- **F**oreign **N**etwork (FN), **F**oreign **A**gent (FA)

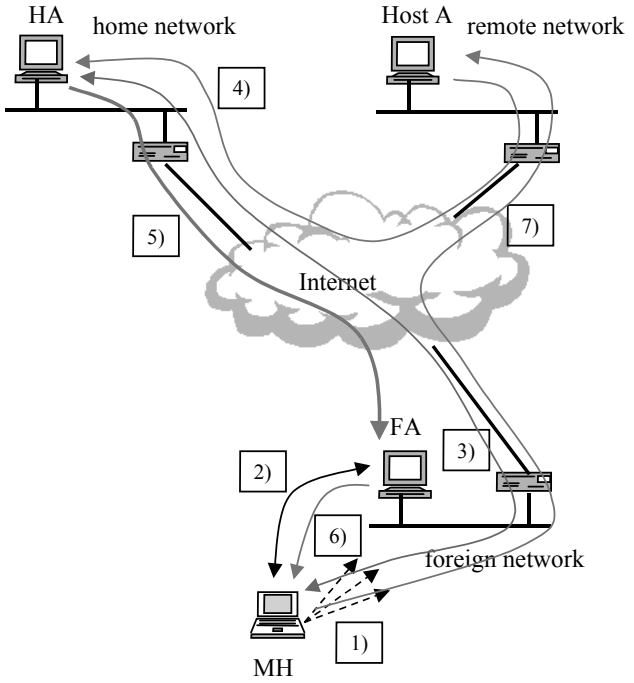
Per una comunicazione
HA manda un pacchetto al CoA
incapsulamento **IP-in-IP**
CoA deve rispondere con l'indirizzo di MH



IP-in-IP encapsulation

scenario di IP mobile

1. Il nodo MH si sposta nella rete FN dove si registra
2. acquisisce un agente FA (1 e 2)
3. si aggiorna la HA al CoA (lo stesso FA)
4. host A contatta Ha per l'host mobile
5. HA fa da tunnel per il CoA
6. FA riceve il pacchetto e lo invia a MH
7. la risposta viene inviata direttamente



Problemi di triangolazione per **ogni messaggio**
In IPv6 si fa caching e si supera il problema

Oltre al normale routing

Estensioni verso la qualità di servizio

Si considera come si possa intervenire sul routing per ottenere garanzie (RFC1889)

i **flussi** sono stream di byte e

si deve mantenere il **flusso con garanzie**

per diverse flussi di traffico, controllando il traffico stesso

Nuove organizzazioni per **qualità pensate per località**

Due organizzazioni diverse:

- **Servizi** organizzati per ogni **singolo flusso** (**Integrati**)
- **Servizi** organizzati raggruppando **flussi** (**Differenziati**)

Una **località** è costituita da diversi nodi interni e da nodi di confine

I **nodi di confine** fanno da **condizionatori** di traffico misurano, marcano, aggiustano il traffico

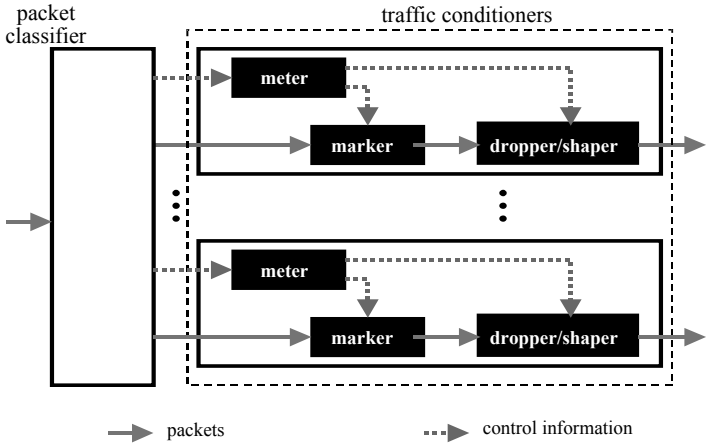
Misurazione del **profilo** di traffico

uso di profili: in-profile, out-of profile

per decidere come trattare il traffico

anche Re-marking (nuovi DS codepoint)

o Shape/drop packets



Traffic Management

Per un buon servizio, è necessaria la gestione del traffico fatta dai nodi router che si occupano del traffico stesso

Router devono gestire **code** e **traffico**

Scheduling e **queue management**

il router deve mandare i pacchetti per i flussi al momento giusto mantenendo QoS

Router devono avere **stato per differenziare i flussi**

Sono necessarie forme di gestione delle code

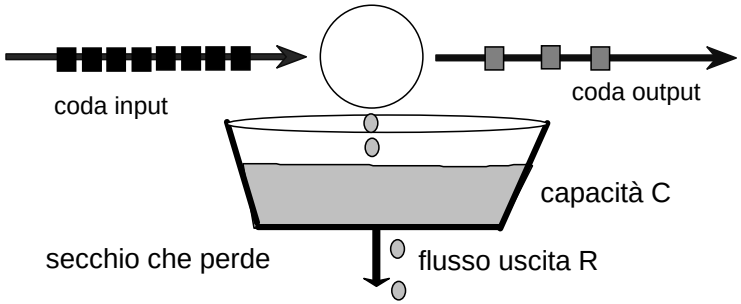
Le **prime forme** sono

Il normale modi di lavoro è il **FIFO**, con una unica coda per tutti i flussi: questo nega qualunque servizio differenziato

il router passa i datagrammi senza considerazione della loro lunghezza o destinazione/sorgente

LEAKY BUCKET

il router scarta i datagrammi considerando la capacità del router e il flusso



Forme di Queue Scheduling

- **FCFS**
- **FQ**
- **BRFQ**
- **WFQ**

Fair Queuing
una coda per ogni flusso, code con uguale priorità

Bit Round Fair Queuing
una coda per ogni flusso, code con uguale priorità
ma con gli stessi byte trasmessi: un flusso con pacchetti grandi viene ritardato rispetto ad altri piccoli

Weighted Fair Queuing
una coda per ogni flusso, code con peso diverso

Altre forme di prevenzione della congestione

scarto random di pacchetti, prima che possa arrivare la congestione

RED - RANDOM EARLY DETECTION

Ci sono molte variazioni
i pacchetti sono scartati in modo random tanto più quanto le code si allungano

RED definisce **lunghezza minima e massima** e **media**
se **coda < minima** nessuna azione
se **coda > massima** nuovi pacchetti scartati
altrimenti scarto con probabilità crescente con la lunghezza della coda

Servizi Integrati INTSERV (RFC2210)

Supporto al QoS a livello N

L'idea dei **servizi integrati** è quella di definire e mantenere un certo **livello di servizio per uno specifico flusso** in un certo **dominio di amministrazione** o anche in uno **scenario globale, sia best effort, sia real-time**

Una applicazione richiede un certo livello di servizio usando una **interfaccia** opportuna e un **protocollo** di **segnalazione**
Il supporto verifica che il servizio si possa fornire (**controllo di ammissibilità**) e accetta di fornirlo

Del protocollo si devono occupare i livelli bassi (di rete)
nel caso di **servizi integrati**
Le applicazioni non si occupano direttamente del protocollo la cui garanzia deve essere ottenuta a basso livello

La specifica del Traffico è fondamentale per accettare e controllare, tipicamente

r	**velocità**	byte/sec
b	**dim. bucket**	byte

Solo dopo la richiesta si comincia il servizio

che deve essere continuamente monitorato per evitare che si usino risorse non riservate

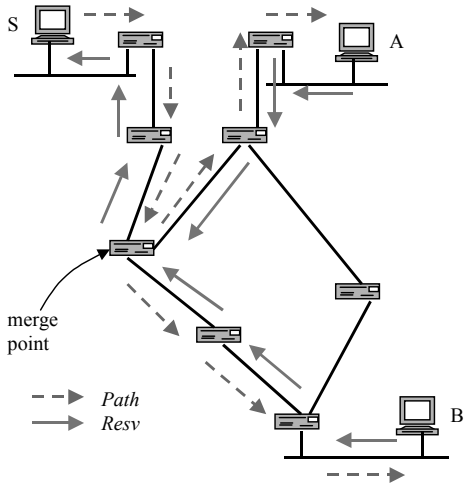
Per considerare uno standard possibile
RSVP Reservation Protocol
Il protocollo riserva le risorse in modo del tutto separato dal traffico corrente sui canali

RSVP (RFC 2205)

Il ReSerVation Protocol provvede alla
gestione (signalling)
user-to-network e network-to-network
attraverso informazioni di traffico *FlowSpec*:

- ***TSpec*** *(descrizione del traffico)* inviate sulla rete
- ***AdSpec*** (opzionale) conferma la reservation al ricevente
- **riservando** in modo **unidirezionale**

RSVP Protocollo a due passi, con soft-state:
sender: *Path* message
nodi mantengono il **soft-state** fino al prossimo *Resv, PathTear* o time-out
receiver: *Resv* message - TSpec (+RSpec)
sender: *PathTear*
receiver(s): *ResvTear*
refresh del soft-state usando ulteriori *Path* e *Resv*



Problemi connessi al riservare

RSVP introduce l'idea di riservare risorse
Il protocollo a due passi
Una reservation può bloccarne un'altra *ResvErr*

Lo **stato** deve essere mantenuto **per ogni ricevente** e si produce traffico per ogni rinfresco dello stato

Inoltre, si possono fornire solo livelli di servizio compatibili per riceventi diversi

Inoltre, ci sono eventi da considerare:
Router failure
QoS può anche degradare fino a best-effort =>
necessario rinegoziare QoS
Applicazioni e router devono sapere che si usa RSVP
problemi con applicazioni legacy

Al momento viene raccomandato
solo per reti locali ristrette
e non per ambienti globali

I limiti suggeriscono anche altri approcci

Supporto al QoS a livello applicativo

Si usa il solo protocollo UDP su questo si costruiscono nuovi protocolli a livello di singolo flusso

RTP => Real-Time Protocol

RTCP => Real-Time Control Protocol

che non garantiscono QoS ma la rendono possibile attraverso una accresciuta visibilità a **livello applicativo**

I messaggi sono mandati in banda

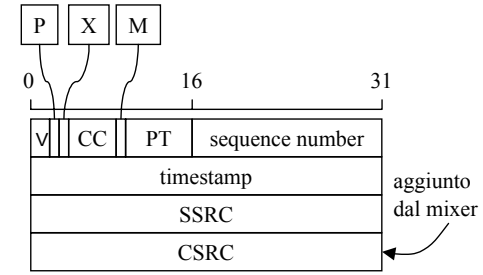
RTP messaggi di marking del traffico e applicativi

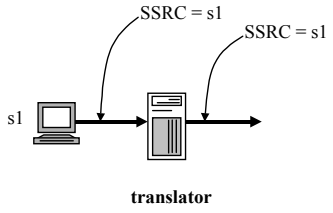
RTCP messaggi di gestione della connessione astratta

Real-Time Protocol

Ruolo **attivo** sia per il **sorgente** sia **mescolatori** (mixer) intervengono nel protocollo

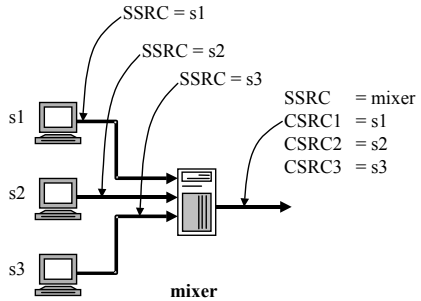
Gli intermediari devono lasciare traccia ed intervenire sul messaggio, per consentire di mantenere le garanzie





V	2-bit, numero versione (=2)
P	1-bit, padding
X	1-bit, indica extension di header
CC	4-bit, numero di CSRC (CSRC count)
M	1-bit, marker specifico per profilo
PT	7-bits, payload type, specifico del profilo
SSRC	synchronisation source
CSRC	contributing source

timestampin un unità specifiche di profilo/flusso



Real-Time Control Protocol

deve fornire informazioni di controllo per un flusso di dati

QoS per flusso

informazioni pacchetti: perdite, ritardi, jitter

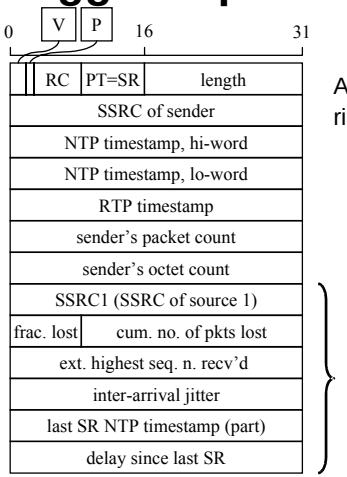
informazioni end system: utente

informazioni applicazione: specifiche di flusso applic.

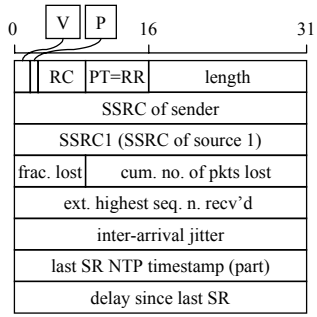
uso di messaggi tipati

RR / SR	**R**eceiver / **S**ender **R**eport
SDES	**S**ource **D**escription
BYE	Abort di sessione
APP	Specifica di applicazione

Messaggi di tipo RR e SR



Anche più istanze ripetute in un report



messaggi di controllo

SDES

Source **DES**cription stringhe ASCII

- CNAME: canonical identifier (mandatory)
- NAME: user name
- EMAIL: user address
- PHONE: user number
- LOC: user location, application specific
- TOOL: name of application/tool
- NOTE: transient messages from user
- PRIV: application-specific/experimental use

BYE

BYE consente di lasciare una sessione RTP

SSRC (o SSRC e lista CSRC se mixer)
reason for leaving

APP

APP definisce pacchetti application-specific

SSRC (or SSRC e lista CSRC se mixer)
ASCII string for name of element
application-specific data

Protocolli di Streaming RTSP

Real Time Streaming Protocol

integrazione di uno streaming acceduto via Web (RealPlayer) dopo avere scaricato la specifica del file
Il player contatta il server via UDP o TCP cercando di ottenere il migliore adattamento tramite buffering
se **UDP** aspetta 2-5 secondi e poi comincia a mostrare
se **TCP** deve usare un buffer più ampio

Servizi Differenziati (DIFFSERV)

L'idea è di **differenziare i servizi** offerti in **classi diverse** con caratteristiche di scalabilità
I **servizi differenziati** sono lasciati ad un dominio specifico di applicazione e un gruppo di IETF sta definendone diversi
I servizi sono a livello di utenti e di comunità di utenti
e di utilizzo più facile degli INTSERV ed adatti per applicazioni legacy

I pacchetti sono marcati a **livello di rete** (non a livello applicativo) e sono riconosciuti e trattati dai router
NON si lavora per ogni flusso di informazioni, ma aggregando classi di flussi

Si usano classi di servizio: come

- **premium** (basso ritardo)
- **assured** (alta velocità, bassa perdita di pacchetti)

ma anche

- **oro**
- **argento**
- **bronzo**

La **classificazione** viene fatta all'ingresso del pacchetto sulla base del contenuto del pacchetto stesso

Service Level Agreement (SLA)

Politica di servizio concordata tra utente e server, e servizio fornito dalla rete con politiche assicurate dai router

DIFFSERV

si possono usare molti modi per differenziare i servizi
ma il più praticabile sembra essere il byte **DS**
nell'header di ogni pacchetto (*ToS in IPv4*)

packet marking nel **DS byte**
- IPv4 ToS byte
- IPv6 traffic-class byte

classificatori di traffico basati su
- multi-field (MF): DS byte + other header fields
- aggregazioni di behaviour (BA): solo DS field
- **DS codepoint** dipendenti dalla applicazione

Si tentano Per-hop behaviour (PHB):
- aggregando flussi nella rete

I **classificatori** di traffico lavorano nella selezione dei pacchetti sulla base delle informazioni contenute negli header, nel modo più ampio possibile
Si possono anche considerare
- le porte,
- il tipo di protocollo,
- il tipo di reservation, ...

Però DIFFSERV presentano ancora limiti rispetto a quello che si può ottenere con RSVP e i servizi integrati

Alcune Proposte

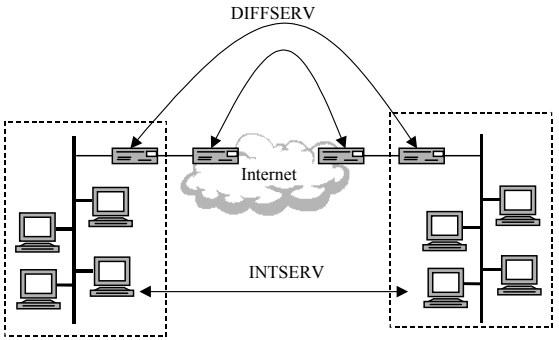
IETF DiffServ sta definendo due **P**er-**H**op **B**ehaviour
- **expedited forwarding**
- **assured forwarding**

Nel caso **Expedited PHB**
- bassa perdita, basso ritardo, basso jitter

Si crea una connessione punto a punto
tipo **virtual leased line** tra endpoint
- *i router devono inoltrare i pacchetti in una classe di priorità che garantisca di utilizzare il massimo della banda e delle risorse disponibili*

Nel caso **Assured PHB**
- distingue quattro classi di servizio
- ciascuna con tre priorità di scarto
- *Solo in caso di congestione, i router cominciano a scartare secondo priorità*

INTSERV e DIFFSERV



Al momento sono in fase di sviluppo sia i protocolli di tipo differenziato, sia di tipo integrato
anche se i **servizi differenziati** sembrano essere più **scalabili** e fornire prestazioni anche a **servizi legacy**
Naturalmente, i **router** devono fornire i nuovi servizi