Università degli Studi di Bologna
Dipartimento di Informatica – Scienza e Ingegneria (DISI)
Scuola di Ingegneria

Corso di **Reti di Calcolatori M**

Variazioni sulla Qualità di Servizio (QoS) e protocolli per la nuova Internet

Antonio Corradi

Anno accademico 2014/2015

Qualità di Servizio per Stream

Molti indicatori e **parametri** per connotare **un flusso di informazioni** e le sue **proprietà funzionali**

Prontezza di risposta

ritardo, tempo di risposta, jitter (variazione ritardo di consegna)

Banda **bit o byte al secondo (per applicazione e sistema)**

Throughput **numero di operazioni al secondo (transazioni)**

Affidabilità **percentuale di successi/insuccessi**
MTBF, MTTR

Aspetti **funzionali** (misurabili facilmente) e **non funzionali**

Molti aspetti legati alla **qualità del servizio** sono **non funzionali** ma dipendenti dalla struttura intera del sistema e della applicazione specifica o da fattori esterni e valutabili solo dall'utente finale

QoS: altri INDICATORI

Per **l'utilizzatore** sono **molto significative** anche proprietà **non funzionali**

- **dettagli dell'immagine**
- **accuratezza dell'immagine**
- **velocità di risposta alle variazioni**
- **sincronizzazione audio/video**

la QoS può essere garantita solo attraverso un **accordo negoziato e controllato**

- osservando il **sistema in esecuzione** e adeguando **dinamicamente** il servizio alle condizioni operative correnti del sistema e dell'ambiente in base alle specifiche dell'**utente**
- **necessità di osservazione e retroazione**

QoS INDICATORI Utente

Le proprietà richieste dall'utente finale tipicamente no funzionali possono essere:

QoE (Quality of Experience)

- **Importanza (priorità)**
- **QoS percepita (dettagli, accuratezza, sincronizzazione e qualità audio/video)**
- **Costo (per accesso, per servizio)**
- **Sicurezza (integrità, confidenzialità, autenticazione, non ripudio)**

QoS deve tenere conto di tutti gli aspetti ai diversi livelli del sistema e tenendo conto di tutti i requisiti

L'**accordo negoziato deve essere verificato durante la esecuzione** per potere fare in modo **veloce azioni correttive**

Qualità di Servizio

Banda (**throughput**) *quantità di dati trasmessi su un canale con successo (per secondo)*

Ethernet 10Mbps (*quantità informazioni/sec*) 10 Mbit al secondo

Tempo di latenza *tempo impiegato per trasmettere una unità di informazione (bit)*

*anche tempo di andata/ritorno (**R**ound **T**rip **T**ime o RTT)*

$$T_L = T_{prop} + T_{tx} + T_q$$

T_{prop} dipende dalla **velocità** della luce nel mezzo (Spazio / Velocità)

T_{tx} dipende dal **messaggio** e dalla **banda** (Dimensione / Banda)

T_q dipende dai **ritardi** di accodamento in diversi punti intermedi

T_q tempo critico che tiene conto dell'overhead

Qualità di Servizio

Un buon servizio richiede di identificare i **colli di bottiglia** e deve considerare *la gestione delle risorse*

se invio/ricezione di 1 byte ⇨ dominante la latenza **RTT**

se invio/ricezione di molti MegaByte ⇨ dominante la banda

Impegno risorse Prodotto **Latenza x Banda**

risorsa canale dati

latenza 40ms e banda 10Mbps → prodotto 50 KB (400 Kb)
è necessario che il mittente invii **50KB** prima che il primo bit sia arrivato al destinatario e **100KB** prima di una risposta al mittente

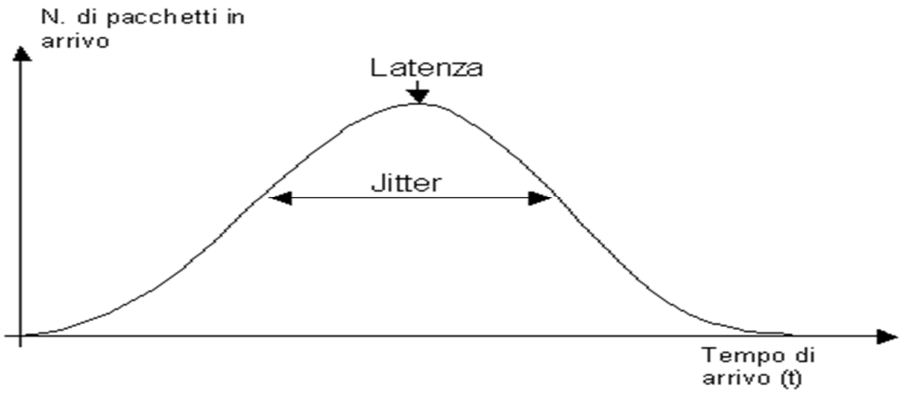
Alcune strategie semplici

Le infrastrutture tendono a mantenere le pipe piene con proprie risorse per la garanzia i tempi di risposta, **ma i tempi vanno sempre considerati**

Si tende ad incorporare un **tempo di buffering nelle applicazioni**

Qualità di Servizio - Jitter

JITTER definito come varianza della latenza in un flusso
situazione ottimale se la latenza fosse fissa ma…



A volte è significativo anche lo **SKEW** come eventuale sfasamento tra più flussi visti come un unico stream (esempio, in uno stream audio / video)

Interesse alla QoS

In caso di sistemi multimediali, o di erogazione di flussi continui di informazioni

Video on Demand (VoD) erogazione di servizi video via una infrastruttura Internet compatibile

perché interesse?

stream di informazioni audio e video con cui giocano fattori real-time: banda, ritardi, jitter, *variazioni di ritardo ammissibile*

Le entità negoziano certe caratteristiche di qualità per i servizi ripetuti o di flusso e li rispettano

- ritardo iniziale per assorbire il jitter medio

- scarto dei pacchetti che arrivano oltre un certo ritardo

QoS in DIVERSI AMBIENTI

TCP/IP CON o SENZA CONNESSIONE

le entità comunicano utilizzando le risorse che sono disponibili al momento della azione (dinamico) senza un impegno predeterminato

Il livello di IP è responsabile della semantica best-effort

IN OSI

le entità OSI impegnano risorse e possono anche fornire garanzie, che devono essere rispettate da tutte le entità del percorso

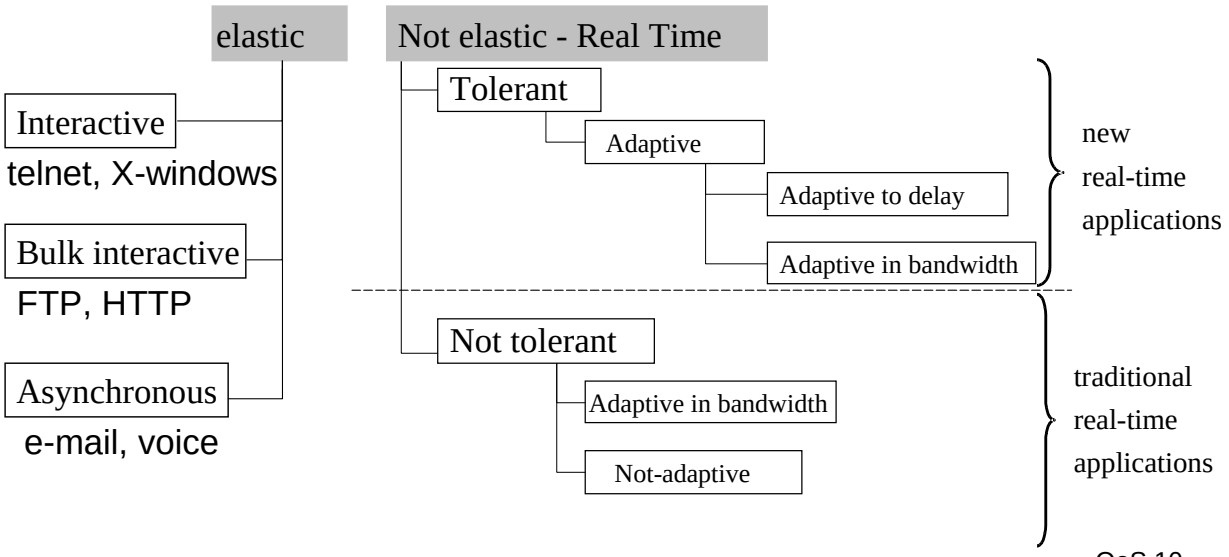
Come garantire QoS in TCP/IP in ambienti best effort?

Applicazioni come i **nuovi servizi applicativi di Internet**

CLASSIFICAZIONE APPLICAZIONI

requisiti di qualità delle applicazioni

Applicazioni Elastiche e Non Elastiche



APPLICAZIONI ELASTICHE o MENO

Le elastiche senza vincoli di qualità ma con requisiti diversi indipendenti da ritardi

lavorano meglio con ritardi bassi e
lavorano male in caso di congestione

Interattive con ritardi inferiori a 200ms

Le non elastiche hanno necessità di garanzie e del rispetto di vincoli di tempo

poco tolleranti per essere usabili al di fuori dello spazio di ammissibilità richiesto

Il servizio si può adeguare ai requisiti

adattative al ritardo → **audio scarta pacchetti**

adattative alla banda → **video che si adatta la qualità**

GESTIONE QoS

La buona gestione si può ottenere con azioni che devono essere attive per l'intera durata del servizio

Le azioni devono essere sia preventive (preliminari alla erogazione) sia reattive (durante il deployment)

sia statiche (preventive), sia dinamiche (reattive)

Azioni statiche

decise e negoziate prima della erogazione

Azioni dinamiche

identificate durante la erogazione

Sono necessari dei modelli precisi di gestione

modello di monitor e qualità

GESTIONE QoS: FASE STATICA

Azioni statiche **Prima della erogazione**

specifica dei requisiti e variazioni

Definizione univoca delle specifiche per i livelli di QoS
Si parla di **Service Level Agreement** (**SLA**)

negoziazione

Accordo tra tutte le entità e livelli interessati nel determinare QoS

controllo di ammissione (admission control)

Confronto tra il QoS richiesto e le risorse offerte

riserva e **impegno** delle risorse necessarie

Allocazione delle risorse necessarie per ottenere il QoS considerato

SLA rappresenta l'accordo statico (come descriverlo?)

GESTIONE QoS: FASE DINAMICA

Azioni dinamiche **Durante la erogazione**

monitoring delle proprietà e delle variazioni rispettando la **politica stabilita**

Misura continua del livello di QoS e dei parametri del SLA

controllo del rispetto e **sincronizzazione**

Verifica del mantenimento e della eventuale sincronizzazione di più risorse (video / audio)

rinegoziazione delle risorse **necessarie**

Nuovo contratto per rispettare QoS

variazione delle risorse per mantenere **QoS** e **adattamento a nuove** situazioni

Dopo la rinegoziazione si deve controllare di nuovo il rispetto di SLA

GESTIONE e MONITORAGGIO

Problema del costo della strumentazione per garantire la QoS

Necessità di avere meccanismi di raccolta di dati dinamici e di politiche che non incidano troppo sulle risorse (usate anche dalle applicazioni)

Tutte le corrette gestioni si scontrano con il requisito di impegnare meno risorse possibili

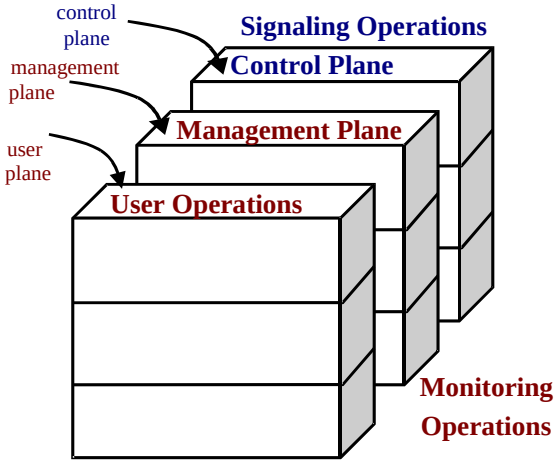
L'area della performance (monitor e gestione dati) deve arrivare a strumenti e politiche che siano meno intrusivi possibile

Principio di minima intrusione

cioè contendendo meno possibile con la applicazione

GESTIONE e MONITORAGGIO

Necessità di **accoppiare il piano operativo** (o utente) con **strategie e strumenti di controllo della operatività**



Piano User

per protocolli utente

Piano di Management

per la gestione durante il servizio
il monitoring,

Piano di Controllo / Segnalazione

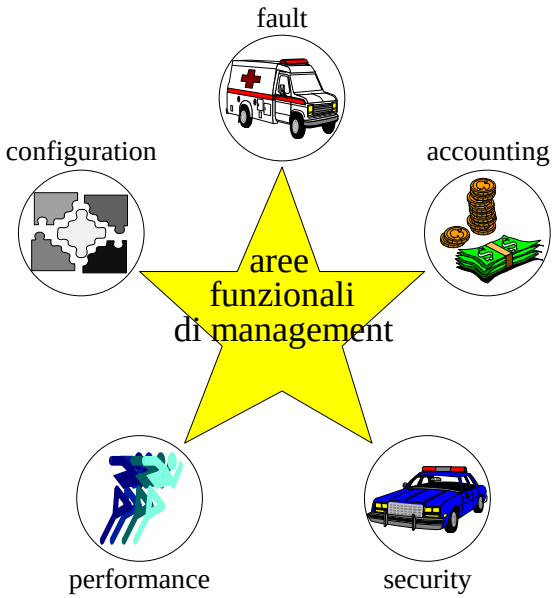
per attuare la connessione, la negoziazione e segnalazione tra livelli non necessariamente in banda (in telefonia, instaura la chiamata)

GESTIONE e MONITORAGGIO

Aree funzionali di Management dei diversi Standard di gestione

- **Fault Management**
- **Configuration Management**
- **Accounting Management**
- **Performance Management**
- **Security Management**

Vedi OSI di ISO
SNMP di IETF
TINA di CCITT



GESTIONE di SISTEMI - OSI

Management Standard OSI (standard durevole)

Modello di network management standard con operazioni molto flessibili, dinamiche, e basato su oggetti astratti

Il mappaggio da oggetti astratti a concreti non è standardizzato
ad es. le interfacce utente sono non standard ma standard de facto

OSI Distributed Management
Uso di descrizione standard di oggetti e azioni
Common Management Information Base (**CMIB**)
Management Information Service (MIS)
Management unico delle informazioni
Common Management Information Service Element (**CMISE**)

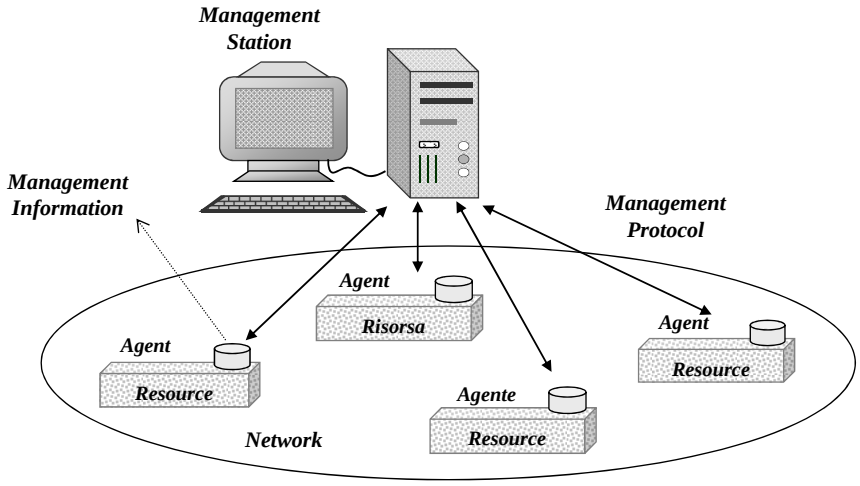
OSI più sofisticato (rispetto al management TCP/IP) si applica a qualunque sistema distribuito per la gestione distribuita delle risorse

NETWORK MANAGEMENT

Management Standard basato su ruoli

- **manager** e
- **agent** che sono responsabili delle **risorse** gestite

Il modello non impone vincoli a priori e può portare a realizzazioni molto semplici o più complesse



GESTIONE di SISTEMI - SNMP

Management Standard IETF

definizione di un **semplice protocollo di management**

SNMP **S**imple **N**etwork **M**anagement **P**rotocol

usando **TCP/IP** e usato in ambienti **UNIX** e **LAN**

SNMP opera su un sottoinsieme di CMIP

incompatibile con lo standard CMIP

con variabili che gli agenti controllano in lettura e scrittura

SNMP è passato attraverso molte ridefinizioni e reingegnerizzazioni

per tenere conto di esigenze di **sicurezza**

per tenere conto di modelli di **gestione flessibili**

per tenere conto di sistemi **legacy esistenti**

…

e per potere gestire non solo apparati, ma **entità di qualunque tipo**

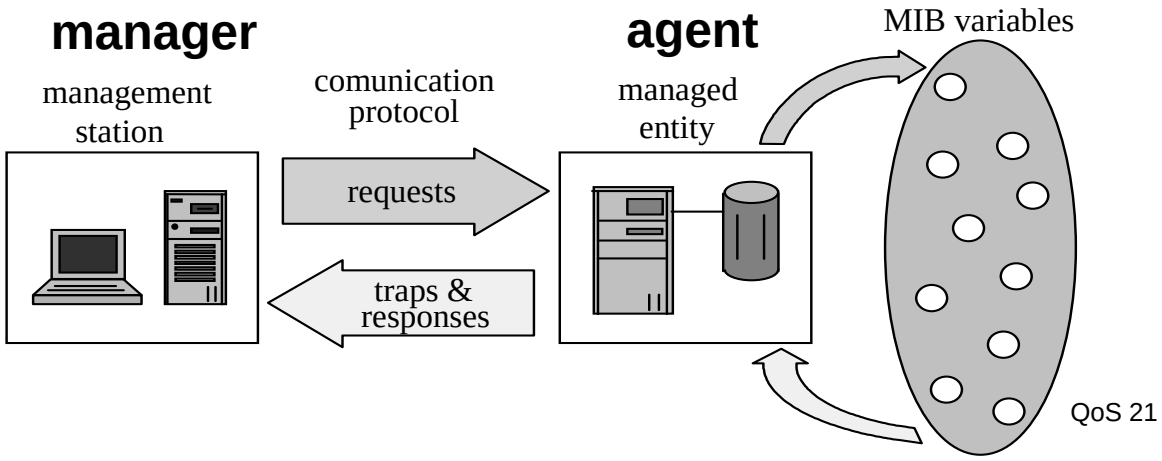
Simple Network Management Protocol

SNMP **S**imple **N**etwork **M**anagement **P**rotocol

Si considera un **manager** (*solamente uno*) e degli **agenti** *(predefiniti)* che controllano **variabili** che rappresentano gli oggetti identificati da nomi unici (OID in directory gerarchici)

Il manager richiede operazioni (*get* e *set*) e riceve risposte

Gli agenti attendono richieste e possono anche inviare *trap*

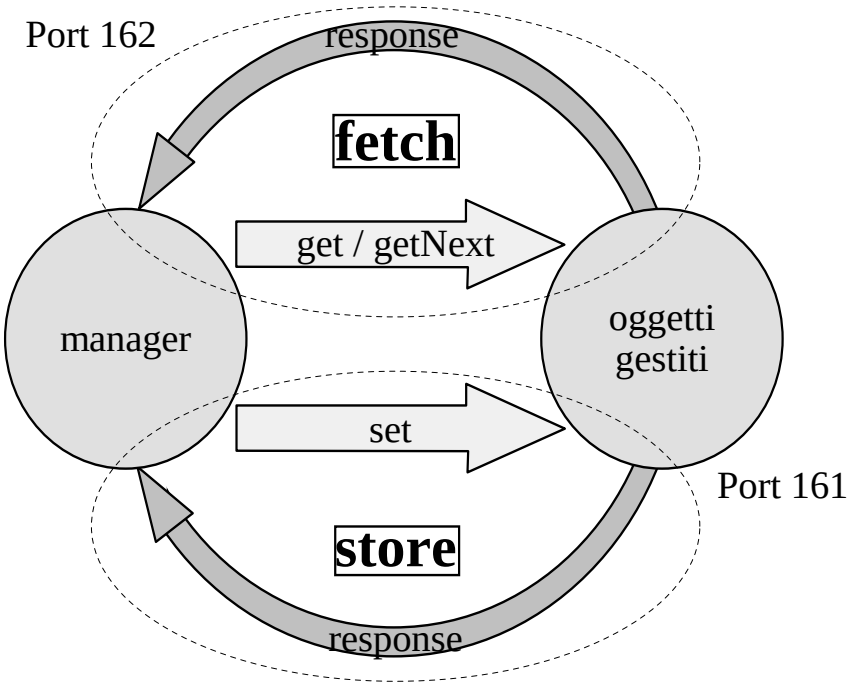


Simple Network Management Protocol

Si usano messaggi molto semplici e limitati
Set,
Get,
Get_Next
(attributi multipli),
Trap
Indicazioni semplici

Uso di UDP
Porta 161 messaggi
porta 162 nel manager per trap

Quali grandezze si possono controllare?
Solo poche

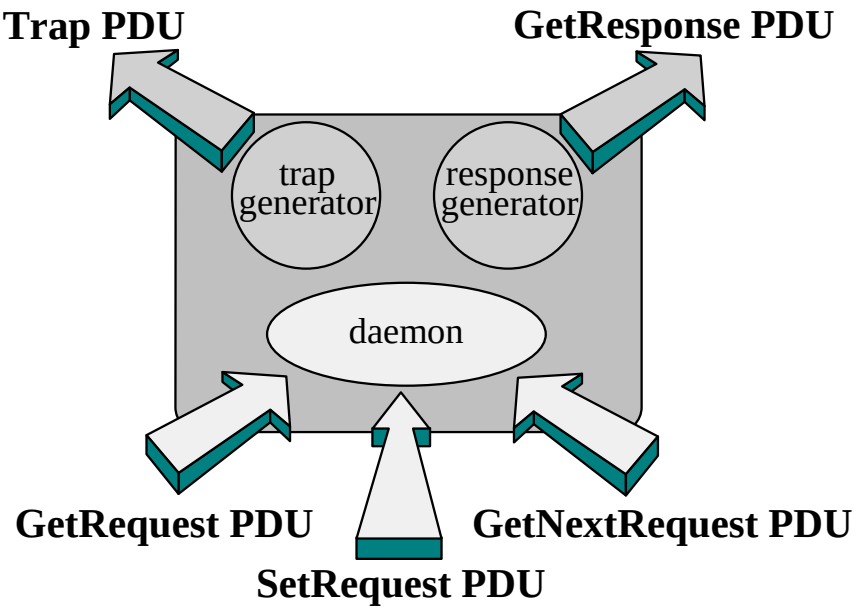


SNMP - Agente

Struttura di un **agente SNMP**

Arrivano richieste di azioni di get e set del manager

Si possono generare trap a fronte di eventi

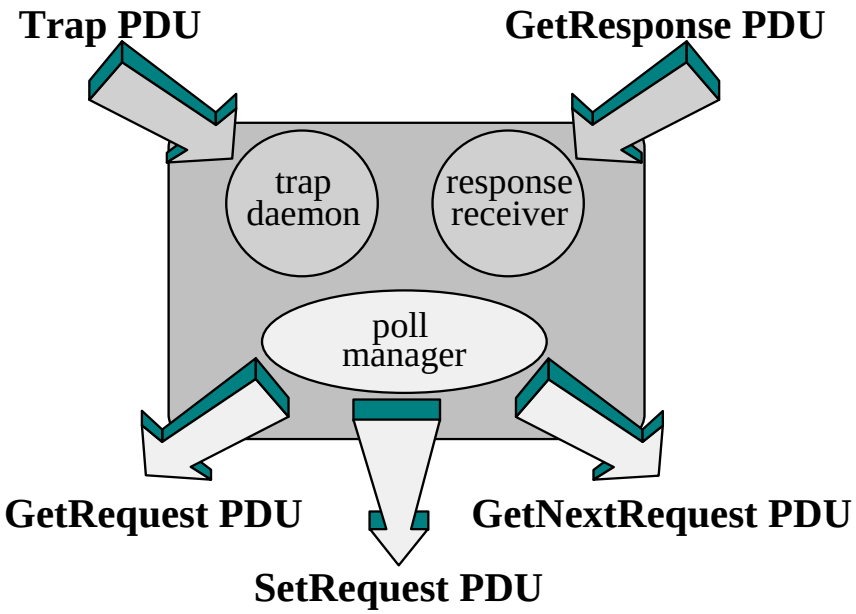


SNMP - Manager

Struttura di un **manager SNMP**

Su richiesta arrivano risposte per le azioni di get e set dagli agenti

Si devono gestire i trap degli agenti



SNMP - STANDARD

SNMP deve consentire la **comunicazione dal manager** all'**agent** in **modo standard attraverso pacchetti standard**

Uso di **descrizione dei dati** via

SMI (Structure of Management Information)

MIB (Management of Information Base)

Entrambe standardizzate in modo preciso

SMI definisce le regole per i nomi degli oggetti (ASN.1 e BER) e **MIB** la collezione di oggetti, tipi e relazioni (secondo OSI X.500)

SMI potrebbe specificare che si scambiano *3* interi ciascuno *a 32 bit* e MIB che stiamo riferendo un oggetto che si trova in un direttorio preciso (1.3.6.1.2.1.7.1 datagrammi IP UDP nel direttorio di base di IETF)

PROBLEMI DI SNMP

SNMPv1

Estrema semplicità e Limitata espressività

Solo aree di **configuration** management (**fault**)

Limitata previsione dei trap (azioni iniziate dall'oggetto)

SNMPv2

Superamento del C/S con gerarchia di manager agent

SNMPv3

Introduzione della sicurezza **S-SNMP**

si trattano i problemi di integrità delle informazioni (anche stream), masquerading, privacy (prevenire disclosure)

non si trattano denial of service e analisi del traffico

In generale, SNMP incorpora le proprietà di CMIP e CMISE

con una visione molto predeterminata prima della esecuzione e poco variabile durante la operatività

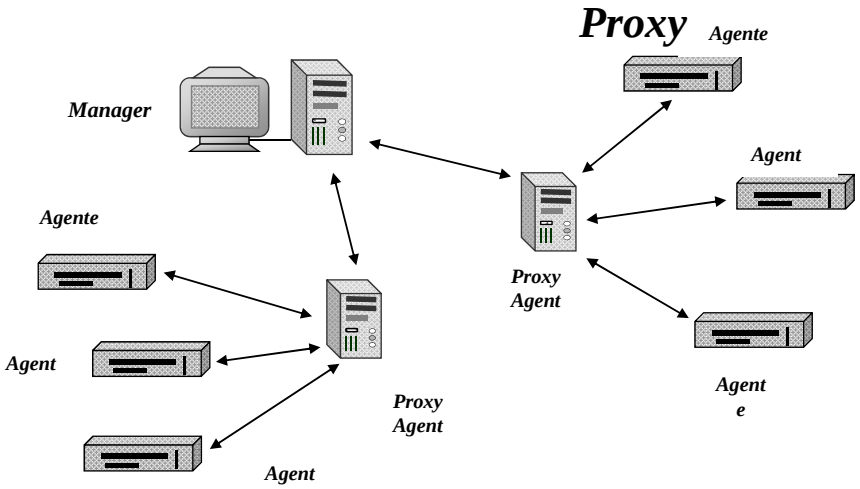
PROBLEMI di CONGESTIONE in SNMP

SNMPv2

Concetto di **agente proxy** ossia **agente e manager**

Entità che si comporta come agent e anche come manager

superando i problemi detti di **micro management**
(*ossia di congestione intorno al manager*)



Il manager comanda una operazione di lettura e i proxy la attuano sulla loro località
Ad esempio riportando i risultati in forma aggregata

GESTIONE RETE e RMON

SNMP gestisce solo grandezze locali agli agenti

Se si deve gestire (il traffico di) rete? Remote **MON**itor

RMON controlla le parti di supporto alla comunicazione ed permette accesso alle statistiche relative

RMON per aumentare la visibilità dell'utente sul traffico

come facciamo a monitorare la rete?

Introduzione di **monitor** e del protocollo di interazione tra **manager** e **monitor**

RMON1 sviluppi nel senso della azioni multiple e innestate

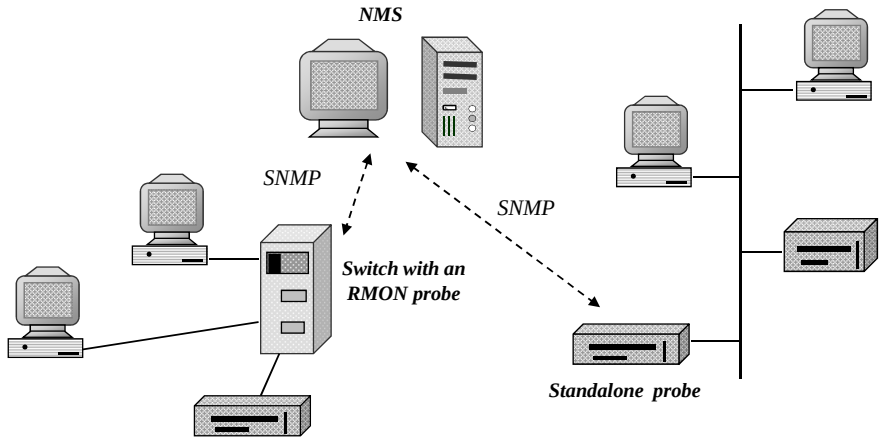
RMON2 e nel senso della garanzia di sicurezza

RMON e PROBE

RMON approccio **orientato al traffico e banda** e non ai dispositivi

probe entità in grado di monitorare i pacchetti sulla rete

I probe *possono lavorare in autonomia* e quindi anche scollegati dal manager fino a *tracciare sottosistemi* e a *riportare informazioni filtrate* al manager

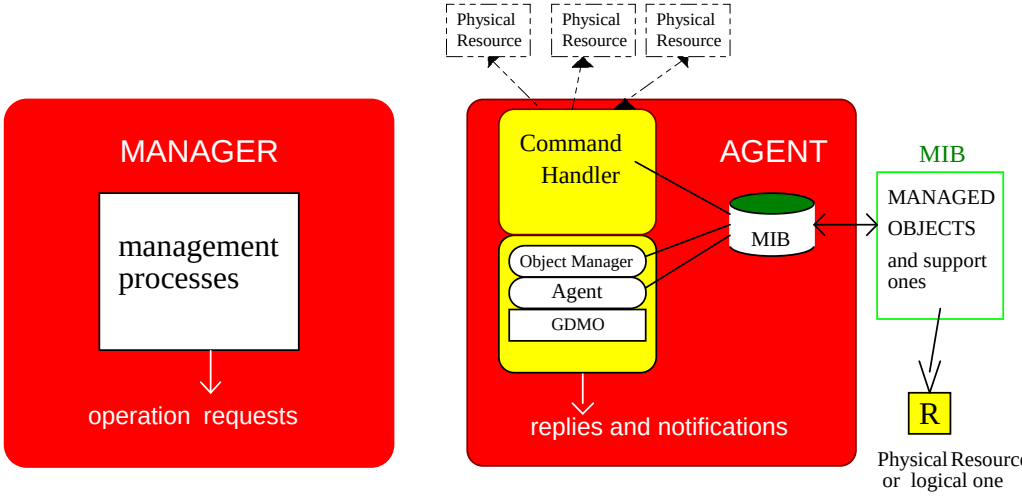


GESTIONE RETE OSI

Modello ampio di **Gestione Distribuita** basato su

entità attive **(manager)**
entità da controllare **(oggetti)**
entità intermediarie **(agenti)**

con oggetti che possono a loro volta essere **manager in gerarchia**



GESTIONE RETE AVANZATA

Managed Object sono le risorse descritte in termini di **oggetti**

Un oggetto astrae una o più risorse nel sistema definendo e permettendo operazioni complesse

risorse semplici *un modem,*

o complesse *più sistemi interconnessi*

… e se ne possono ***creare dinamicamente***

I **Manager** gestori realizzano le **politiche di gestione sulla base di più agenti di loro competenza o di altri manager**

Un manager può inserire una risorsa o toglierla dinamicamente dal sistema di gestione

Gli **Agenti** agiscono su richiesta dei manager per **fornire le funzioni da attuare su comando**

servizi di attuazione comandi, raccolta informazioni

ma anche di inserimento risorse, creazione nuovi agenti

GESTIONE RETE AVANZATA

Management entity usano il protocollo **CMISE/P**

Insieme di operazioni remote per la comunicazione tra manager ed agenti realizzando un modello dinamico al massimo grado

Set-Modify	stabilire, aggiungere o togliere un attributo ad un oggetto
Get / Cancel Get	lettura di un attributo ad un oggetto (e revoca lettura)
Action	azione su uno o più oggetti
Create/ Delete	richiesta di una generazione/distruzione ad un agente
Event Report	invio di un evento notificato dall'agente al manager

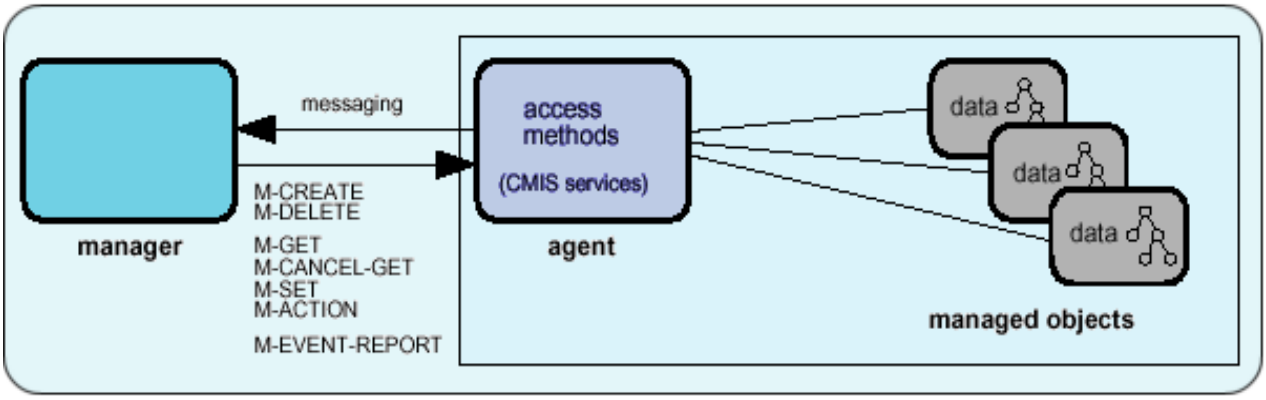
Si noti la aggiunta dinamica *di attributi, azioni, agenti, e eventi per cambiare la struttura del sistema durante la esecuzione*

GESTIONE RETE AVANZATA

Operazioni di Management in **OSI**

per consentire **un controllo dinamico**

operazioni per creare agenti e nuove azioni



GESTIONE e MONITORAGGIO

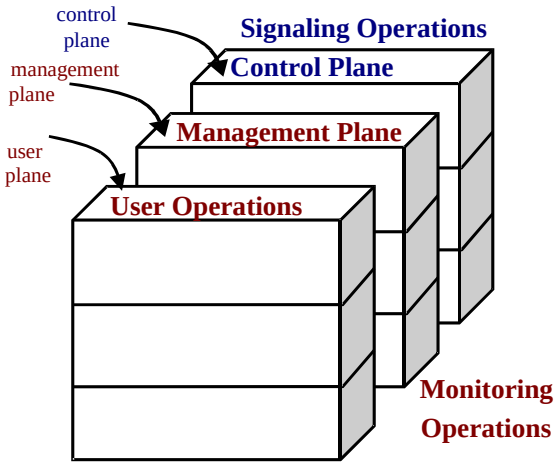
I diversi piani, dal piano operativo agli strumenti di controllo della operatività

Piano User

per protocolli utente

Piano di Management

per la gestione durante il servizio, il monitoring



Piano di Controllo / Segnalazione

per attuare la connessione, la negoziazione e segnalazione tra livelli non necessariamente in banda (in telefonia, instaura la chiamata)

PROTOCOLLO di SESSIONE SIP

Necessità di protocolli **per supportare e gestire le sessioni multimediali,** in modo simile alla gestione delle comunicazioni telefoniche sul piano del controllo

Il protocollo è **Session Initiation Protocol o SIP**

SIP (RFC 2543) è un protocollo recente (1999) e aggiornato, accresciuto, ed esteso (e.g., con eventi - RFC 3261- 2002)

Obiettivo è di **definire e gestire una sessione di supporto ad un servizio multimediale**, attuato poi con altri protocolli

- SIP ha come base la capacità di **segnalazione** per stabilire, modificare o chiudere una sessione multimediale
- SIP si basa su comunicazione di contenuto **HTTP compatibile**
- SIP è un protocollo **text-based** e **cliente/servitore**

MESSAGGI in SIP

I messaggi fondamentali scambiati, l'unica cosa che viene standardizzata, sono appunto messaggi di formato HTML

Pochi tipi di messaggi:

Messaggi di REQUEST

INVITE / ACK / CANCEL / BYE

Altri Messaggi

REGISTER (informazioni di contatto)

OPTIONS Richiesta ai server delle capacità

Messaggi di RESPONSE

PROVISIONAL / FINAL

1xx provisional,

2xx success,

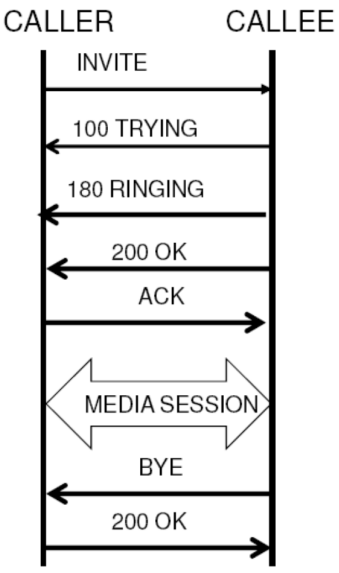
6xx failure

TUTTI i MESSAGGI SIP

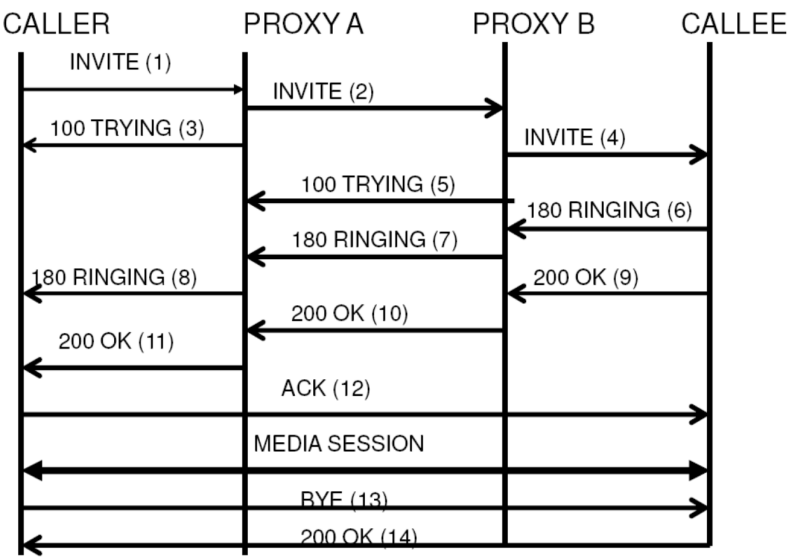
IETF	Nome metodo	Utilizzo
RFC 2976 (2000)	INFO	Per mandare dentro la sessione informazioni che non ne modificano lo stato
RFC 3261 (2002)	INVITE	Per instaurare una sessione, il client invita a partecipare il server
RFC 3261 (2002)	ACK	Il client conferma al server lo stabilimento di una sessione
RFC 3261 (2002)	BYE	La manda uno dei due partecipanti all'altro per terminare una sessione
RFC 3261 (2002)	CANCEL	La manda il client al server per cancellare la sua richiesta pendente
RFC 3261 (2002)	REGISTER	Mappa l'URI del client con la sua locazione corrente
RFC 3261 (2002)	OPTIONS	Richiesta fatta a un server per scoprire le sue capacità
RFC 3428 (2002)	MESSAGE	per trasmettere un messaggio istantaneo
RFC 3262 (2002)	PRACK	per confermare la ricezione di una risposta provvisoria
RFC 3311 (2002)	UPDATE	per modificare alcune caratteristiche della sessione
RFC 3265 (2002)	SUBSCRIBE	un UA si registra a un particolare evento di interesse
RFC 3265 (2002)	NOTIFY	Si notifica un UA di un nuovo evento a cui è interessato
RFC 3515 (2003)	REFER	per istruire un server sul trasferimento di una richiesta
RFC 3903 (2004)	PUBLISH	un UA pubblica un evento

SCENARI SIP

Un caso di base semplice e diretto



Un caso più complesso con diversi intermediari tra cliente e servitore



SCENARI SIP

Si possono anche prevedere delle entità funzionali diverse dai clienti e servitori multimediali, con altre entità coinvolte

User Agent

Endpoint che possono comportarsi come user agent dei clienti (REQUEST) o dei servitori (RESPONSE) per attuare il protocollo

Proxy Server

Router di livello applicativo che possono mantenere o meno stato delle transazioni di sessione (altrimenti stateless), ossia delle richieste mandate da un cliente e delle risposte rimandate al cliente

Redirect Server

Server capaci di rimandare un cliente ad una nuovo server alternativo

Registrar Service

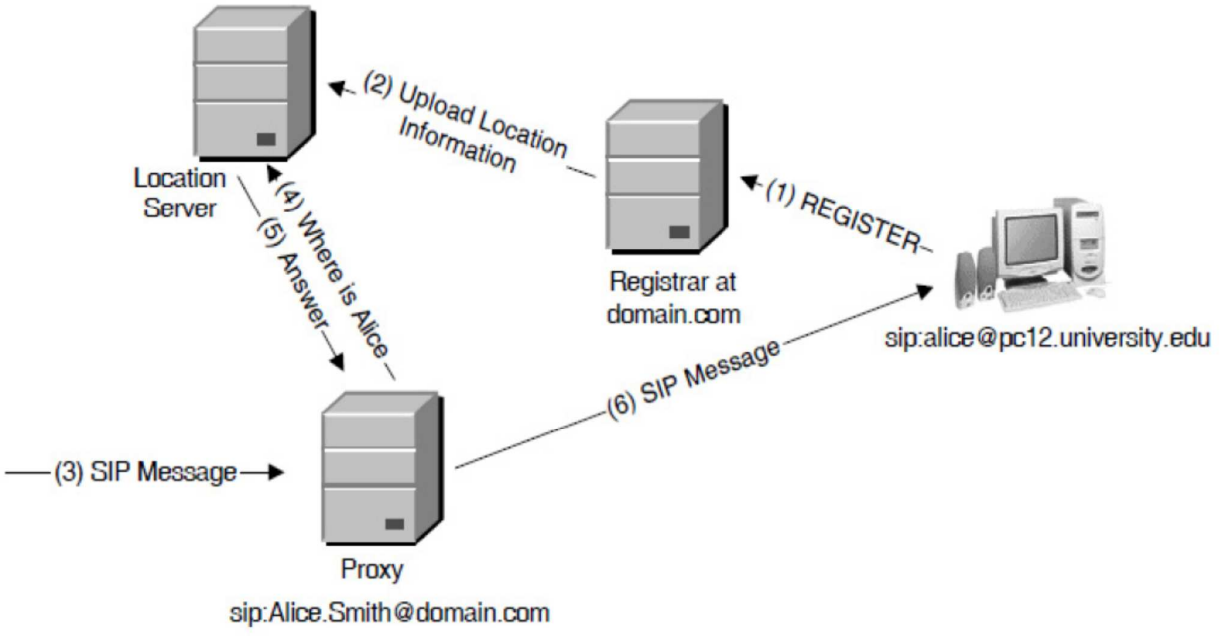
Servizio per la registrazione degli utenti nella infrastruttura

Location Service

Servizio per consentire il collegamento degli utenti interessati alla loro locazione

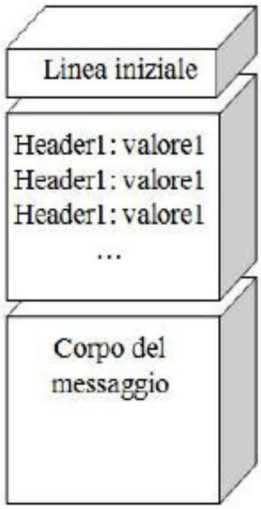
SCENARIO di uso di SIP

Uno scenario un po' articolato
Proxy, Location, Redirect e Registrar Service



STRUTTURA dei MESSAGGI SIP

I messaggi sono tutti costituiti da:
start-line, header, message body (opzionale)



Per i messaggi REQUEST
La request-line come start-line poi
- nome-metodo,
- versione protocollo,
- URI della richiesta

Per i messaggi REPLY
La status-line come start-line poi
- versione protocollo,
- codice stato,
- frase esplicativa

Il body può essere presente per contenere informazioni ulteriori sul flusso e sul servizio

MESSAGGI SIP: INVITE

Un esempio di INVITE

INVITE sip:bob@biloxi.com SIP/2.0 (REQUEST LINE)
Via: SIP/2.0/UDP
pc33.atlanta.com; branch=z9hG4bK776asdhds
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>; tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142
…
Message body (opzionale): una descrizione SDP (Session Description Protocol) per negoziare i formati audio/video

MESSAGGIO SIP

Un esempio di RESPONSE alla request OPTIONS

SIP/2.0 200 OK **(STATUS LINE)**
Via: SIP/2.0/UDP
pc33.atlanta.com; branch=z9hG4bKhjhs8ass877; received=192.0.2.4
To: <sip:carol@chicago.com>; tag=93810874
From: Alice <sip:alice@atlanta.com>; tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 63104 OPTIONS
Contact: <sip:carol@chicago.com>
Contact: <mailto:carol@chicago.com>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE
Accept: application/sdp
Accept-Encoding: gzip
Accept-Language: en
Supported: foo
Content-Type: application/sdp
… Message body (opzionale) …

ESTENSIONI per QoS

Traffic Management

Per un buon servizio, è **necessaria** la **gestione del traffico** tipicamente **attuata** dai **nodi router intermedi** che si devono occupare del traffico stesso (oltre il **best-effort**)

Router devono gestire code e traffico

Scheduling e queue management

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

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

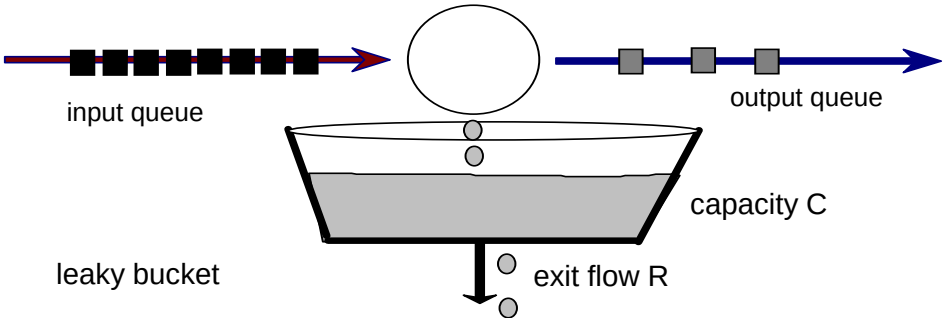
Sono necessarie forme di gestione delle code

ROUTER INTERNET

Il Router passa i pacchetti senza diversificare ***accodamenti*** *o* ***scheduling*** *senza nessuna distinzione tra i flussi*

il router esegue per ogni pacchetto che arriva in **coda FIFO**:

1) Verifica della **destinazione**
2) Accesso alle **tabelle di routing** per trovare un cammino di uscita
3) Selezione del **migliore percorso** in uscita per il pacchetto tenendo conto del **match più adatto** (massima lunghezza di match)
4) Invio del pacchetto attraverso la interfaccia selezionata dal cammino scelto

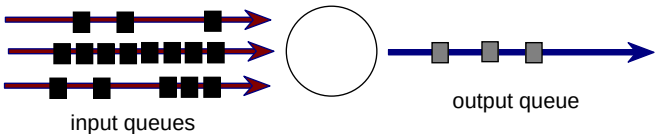


ROUTER INTERNET

Router in Internet best-effort

il router passa i datagrammi senza considerare la lunghezza o gli attributi destinazione/sorgente

Il normale modo di lavoro è **FIFO**, unica coda per tutti i flussi: questo impedisce qualunque servizio **differenziato**



Politica semplice e code unificate

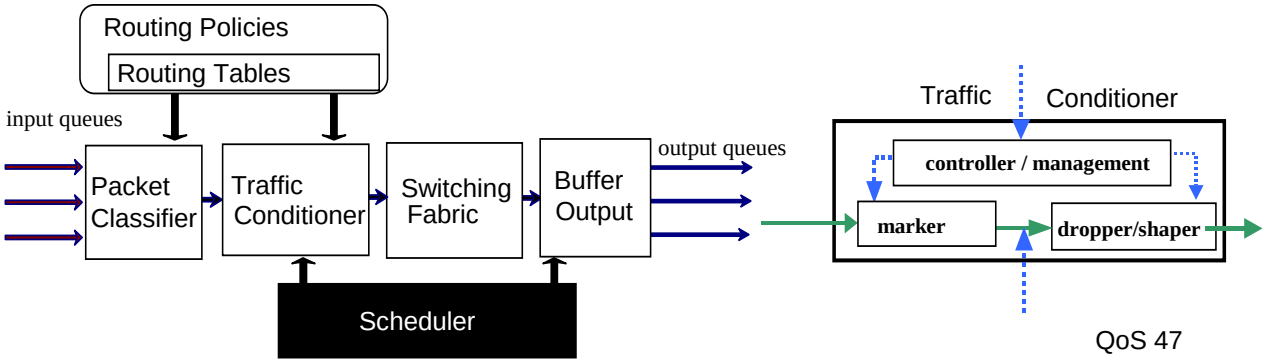
Un pacchetto in uscita (di qualunque lunghezza) potrebbe impegnare il router e bloccare ogni altro flusso

Non possiamo risparmiare risorse per flussi che ne possano avere bisogno

ROUTER per QoS

*Il Router considera politiche per **accodamenti** o **scheduling** basate sulla **lunghezza** o **destinazione/sorgente dei pacchetti** (differenziando i flussi)*

il router può prevedere anche, oltre al **classificatore** dei pacchetti in base al flusso (sorgente/destinazione e lunghezza), anche una funzionalità di **condizionamento del traffico che può decidere anche di buttare via o ritardare pacchetti per alcuni flussi**

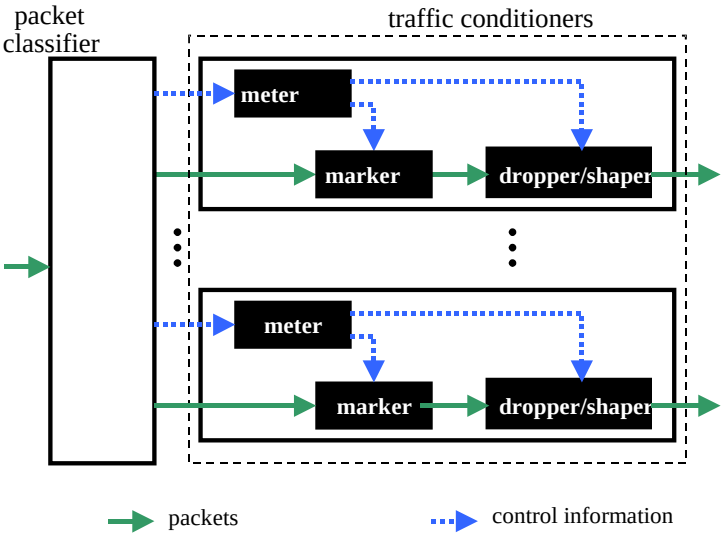


ESTENSIONI per QoS

Il problema fondamentale è come intervenire sul **routing** per ottenere **garanzie** (RFC1889) sui **flussi stream di byte**

Router organizzati sulla base di una effettiva località

località costituita da nodi interni e da nodi di confine



POLITICHE di SERVIZIO per ROUTER

Politiche sui router*: i router devono smistare messaggi*

La prima politica che viene in mente è appena c'è un pacchetto, inviarlo in uscita senza ritardo (**e politica di gestione**)

Questa politica **non introduce ritardi** se non dovuti ad altri pacchetti in uscita e viene definita conservativa del lavoro

(**work conservative o che conservano il lavoro**)

Router best-effort sono conservativi

I router possono lavorare secondo una politica di ***conservazione del lavoro* o meno**

legge di conservazione di Kleinrock: un router (router work-conservative) non può essere idle se ci sono pacchetti da portare in uscita (**non si possono introdurre ritardi sul traffico in alcun modo**)

Quando si considera QoS, i router possono anche **introdurre ritardi** per non penalizzare alcuni flussi: un pacchetto meno prioritario potrebbe essere ritardato anche se non ce ne sono altri adesso (*ma potrebbero arrivarne*)

LEGGE di KLEINROCK

Legge di conservazione di Kleinrock (per router work-conservative): il router non può essere idle se ci sono pacchetti da portare in uscita

*Se ci sono **n flussi** con traffico λ_n per ogni flusso, e se il flusso n ha un tempo di servizio medio μ_n, allora l'utilizzo è dato da* $\rho_n = \lambda_n\mu_n$ *dove*

ρ_n *rappresenta l'utilizzo medio di quel flusso, mentre*

q_n *indica il tempo di attesa medio per il flusso n*

La legge di Kleinrock per **scheduler work-conservative** verifica

$$\Sigma\rho_n q_n = \text{Costante}$$

cioè

si può dare o un **ritardo minore** o una **maggiore banda** a un flusso, **solo se facciamo crescere il ritardo di un altro** o **facciamo diminuire la banda di un altro**

Anche nel rispetto della legge, per favorire un flusso con un router con prestazioni limitate e definite, possiamo solo sfavorirne altri, in condizioni di carico elevate

Router per ottenere **QoS sono anche non conservativi**

MODELLI per ROUTER con QoS

Router con caratterizzazione del traffico

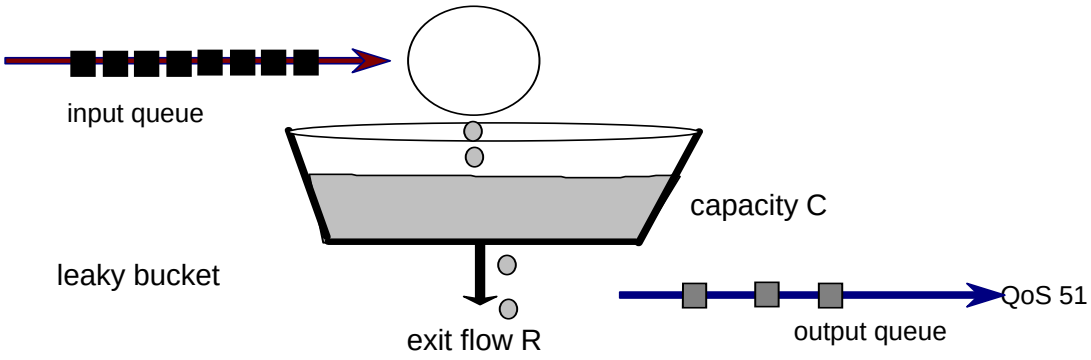
il router deve conoscere i flussi e il servizio possibile *(capacità del router) e deve* **potere gestire** *il traffico*

Modello LEAKY BUCKET per modellare un **servizio ATTIVO del router** e **limitare** il **flusso in uscita (un secchio per flusso/flussi)**

Possiamo controllare dei flussi attraverso la capacità:

Se arrivano dati ***troppo velocemente*** *oltre il flusso in uscita ammissibile, vengono rallentati* (best-effort)

Se arrivano dati ***oltre la capacità*** *vengono persi* (best-effort)



LEAKY BUCKET

LEAKY BUCKET per caratterizzare il traffico

r flusso massimo di uscita, **R** flusso medio di arrivo

LEAKY BUCKET spegne i burst di pacchetti

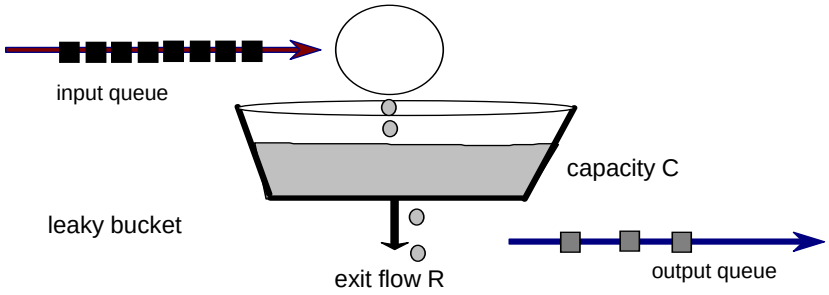
Un pacchetto accodato solo se c'è posto nel secchio (*altrimenti scartato*) e dipende dalla capacità del secchio **C**

I pacchetti possono uscire con velocità massima che dipende dal flusso ammesso in uscita (**r < R**)

Se 100Mbyte in 300msec e se ne fanno uscire a 33Mb/sec, *il leaky regola il flusso portandolo a quello ammissibile*

Se 150Mbyte in 300msec, cominciamo a perdere dati (≅ 50Mbyte)

c = 100Mbyte
r = 33Mb/sec



TOKEN BUCKET (storia del flusso)

TOKEN BUCKET come modellazione del traffico tenendo conto della storia passata dei flussi: il secchio accumula - con regolarità per ogni flusso - token che servono per il passaggio dei pacchetti

I token vengono generati a tempo uniformemente

TOKEN BUCKET permette anche burst di pacchetti

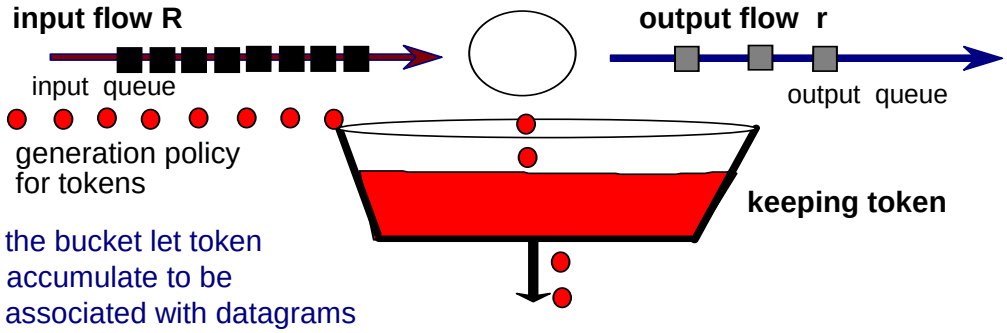
Dati oltre la capacità non vengono persi ma solo ritardati

Se arrivano dati troppo velocemente oltre il flusso in uscita ammissibile, possono anche uscire per accumulo dei token

Se il bucket è **vuoto** → attesa e non si passa

Se il bucket è **pieno** → si possono impegnare tutti i token

Se **parzialmente pieno** → qualcosa può passare, il resto aspetta

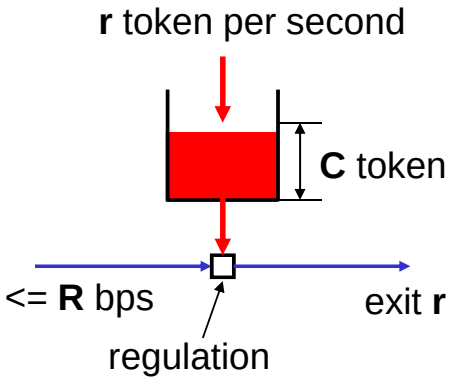


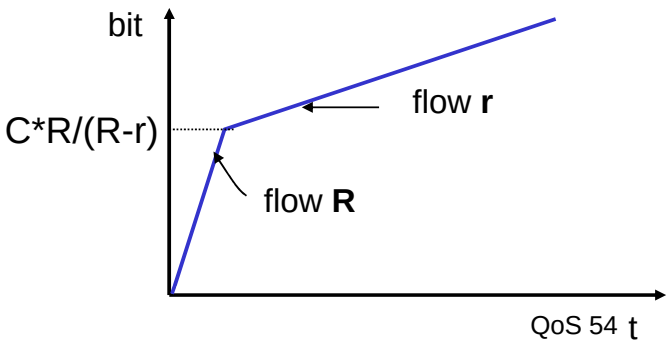
TOKEN BUCKET per QoS

Modello TOKEN BUCKET per il servizio del router con variazioni e politiche diverse

Attesa di un **numero di token** congrui e **invio di tutti i bit insieme** del pacchetto

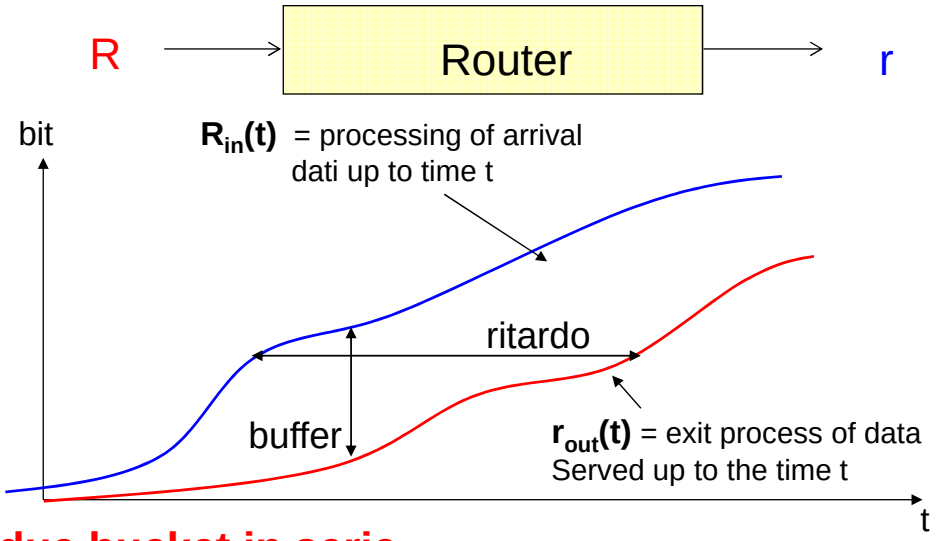
Il pacchetto non viene scartato se non contenibile, ma solo se ne ritarda l'invio (almeno nelle politiche semplici)





SERVIZIO - QoS

TOKEN BUCKET impone vincoli sui flussi, inteso come ritardo, tenendo conto dei **flussi** e richiedendo **risorse di bufferizzazione**, per l'uscita dal router



Spesso i due bucket in serie

POLITICHE per ROUTER con QoS

Router Internet - First Come First Serve o FIFO - lavorano rispettando la legge di conservazione di Kleinrock

invece, per dare priorità ad un flusso si devono sfavorire gli altri
Si tendono a pensare e sperimentare altre politiche

Scheduling e **Accodamento** con rispetto di alcune proprietà

Facilità implementativa

per consentire progetto dei router e realizzabilità effettiva

Giustizia (fairness) e **Protezione**

in condizioni operative uguali nessun flusso deve ricevere meno di altri

Limiti di performance

come vincoli sulla corretta operatività dei diversi flussi

Admission Control

come decisione di ammissione prima della erogazione

POLITICHE FAIR GENERALI: MAX-MIN

PRINCIPIO - Max-Min Fairness

Criterio generale per rispondere alla **proprietà di fairness**, spesso implementato con una **politica più facile da realizzare**

Max-Min share → **le richieste di risorse dei diversi flussi devono essere considerate in ordine di richieste crescenti** (**prima quelli con esigenze minori poi quelli con esigenze superiori**)

C capacità massima globale di risorse
X_n richiesta di risorse del flusso n **$X_1 < X_2 < X_3 < \ldots X_i < \ldots X_{N-1} < X_N$**
m_n risorse allocate al flusso n con successo precedentemente
M_n risorse disponibili al flusso n

$m_n = \min (X_n, M_n)$ e $M_n = \frac{C - \sum_{i=1}^{n-1} m_i}{N - n + 1}$

Si possono anche considerare pesi diversi per i diversi flussi

GENERAL. PROCESSOR SCHEDULING

Il **modello Max-Min** tratta e fa passare prima chi presenta richieste meno gravose, e successivamente gli altri in ordine di peso di richieste … si scala solo se risorse scarse

Generalized Processor Scheduling (GPS)
Modello fluido del traffico

Questo criterio risponde ad un servizio uno alla volta in ordine (Round Robin) molto fair

Ad ogni giro si serve un solo bit per flusso che viene portato in uscita

Si potrebbe dimostrare che questa politica di scheduling è ottima per i servizi

Purtroppo → il **GPS NON** è praticamente implementabile

Si possono servire **solo pacchetti e non bit** (overhead)
Se devono fare delle approssimazioni **facili da implementare**

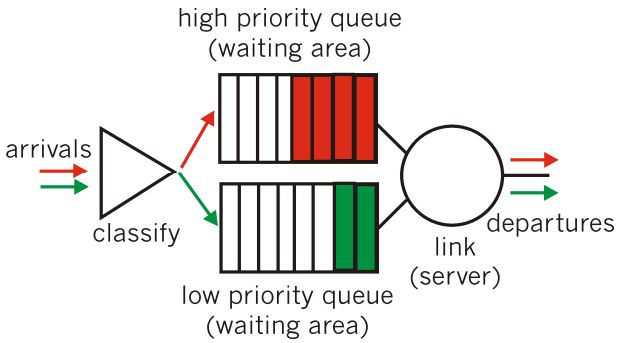
REALI POLITICHE di SCHEDULING

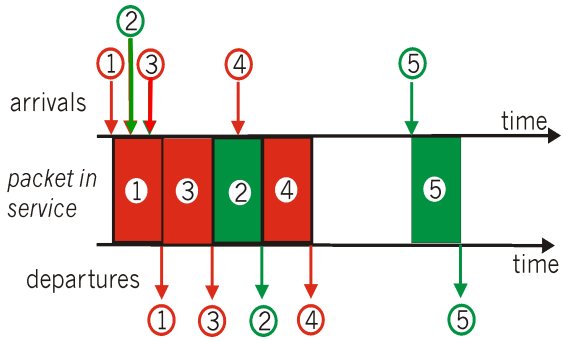
Strategie alternative a FIFO o FCFS

Forme di Queue Scheduling (*tipicamente* ***non work-conservative***) per evitare che un **flusso eccessivo non controllato** possa **congestionare** l'intero traffico e **tutti i flussi**

Scheduling con Priorità

politica di accodamento e scheduling facili da implementare
Un flusso prioritario può causare *starvation* di un flusso meno prioritario

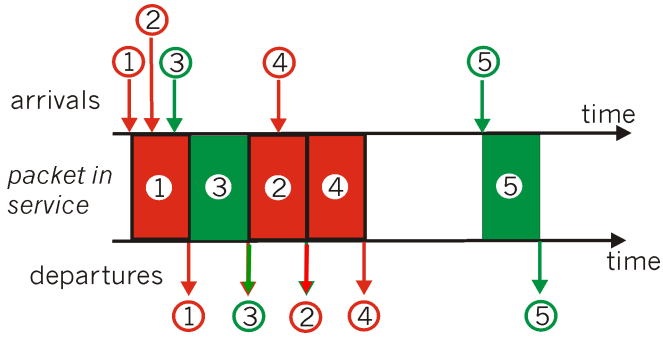




POLITICHE ROUND ROBIN

Round Robin

flussi serviti dalla politica di gestione in round-robin (se traffico)
Serviamo ripetutamente traffico di un flusso se è l'unico presente



Weighted Round Robin

I flussi sono serviti in round-robin in *proporzione a un peso assegnato ad ogni flusso* e ogni coda è visitata per ogni giro un numero di volte pari al peso e per un numero di pacchetti dipendente dal peso

Il peso normalizzato difficile da valutare per flussi corti

ALTRE VARIAZIONI del ROUND ROBIN

Deficit Round Robin

Ogni flusso mantiene un **valore di stato** (**deficit** a zero)
Alla visita della coda, il pacchetto è **estratto se minore di una *certa soglia***, altrimenti non estratto ma registrando storicamente nel deficit la attesa (aumentando il deficit di un quanto per ogni visita)

I pacchetti anche oltre la soglia passano dopo una attesa opportuna proporzionale alla dimensione

Funziona bene per pochi flussi e piccoli pacchetti

Ci sono molte altre variazioni del Round Robin con prestazioni diverse e algoritmi vari di costi diversi

Però *si tende a una visita ed estrazione in ordine…*

SCHEDULING FAIR: FAIR QUEUING

Fair Queuing e sue variazioni

Principio di GPS, come se fosse fatto bit-a-bit

Un pacchetto di un flusso di dimensione N può essere inviato solo dopo avere visitato le altre code N volte con esame di 1 bit per volta

Non si mandano però i messaggi un bit alla volta, ma usando tag di fine messaggio per ogni coda per scegliere il pacchetto che deve uscire per primo

(*quello che avrebbe completato per primo il servizio bit-a-bit*)

Il FAIR QUEUING è la politica più adottata e semplice da realizzare, tipicamente disponibile in tutti i router anche a basso costo

Anche con sue variazioni

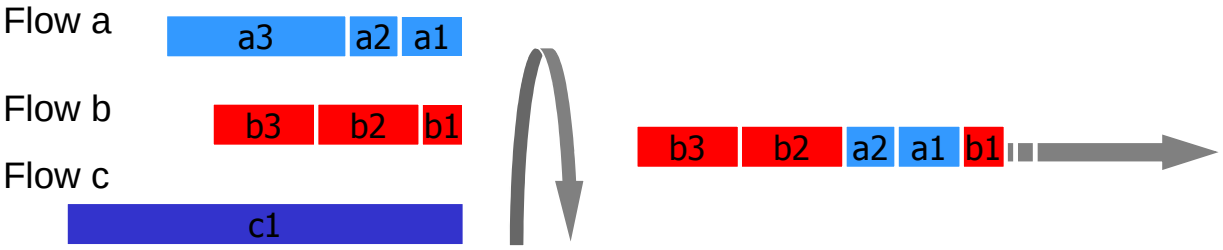
Weighted Fair Queuing con peso diverso associato ai flussi

FAIR QUEUING SCHEDULING

FAIR Queuing

Lo scheduling viene attuato considerando le diverse code e i messaggi in queste

Si fanno uscire **per primi i pacchetti "che finiscono prima"** e che non ostacolano gli altri (scenario di flussi tutti allo stesso livello), cioè *con dimensione che impegna meno il router in uscita*



Si possono anche pesare i flussi in modo diverso (con pesi come in **Weighted WFQ**)

SCHEDULING: problema

Fair Queuing rappresenta una politica giusta che tende a privilegiare un buon uso delle risorse tenendo conto dei vincoli reciproci dei flussi

Ma c'è un problema generale

NON CONOSCENZA in **ANTICIPO** del **TRAFFICO IN ARRIVO**

il problema è che alla trasmissione di un pacchetto non sappiamo cosa stia arrivando sui flussi che contendono le stesse risorse

Soluzione: dove possibile inserire stato nel sistema intero

In caso di **traffico applicativo a flusso**, si può stimare, prima della erogazione, il potenziale impegno delle risorse e quindi tenerne conto come stato di previsione delle risorse

(attraverso le specifiche utente e i costi)

Scalabilità ??? e costi ??? ☹

PREVENZIONE della CONGESTIONE

Una delle situazioni più spiacevoli dei ***sistemi best-effort***

Congestione in cui nessuno lavora più correttamente
Spesso affrontata con **politiche semplici e reattive**

In Internet tradizionale best-effort
si possono fare **solo azioni reattive**

- scartare solo i pacchetti in eccesso (*in modo silenzioso*)
- oppure mandare indicazioni di limitare il traffico (*pacchetti choke*)

Nella nuova Internet con QoS con varie strategie
si possono **fare anche azioni preventive**

- Ad esempio un *uso della finestra di trasmissione* su un canale
- o altro che *prevenga situazioni pericolose*

POLITICHE PROATTIVE: RED

RANDOM EARLY DETECTION (o RED)

una coda per ogni flusso, e *code con uguale priorità*

prevenzione della congestione con uno ***scarto random*** dei pacchetti di ogni coda, anche molto prima di arrivare alla congestione

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

RED definisce lunghezza minima e massima e media di coda

Se coda < **soglia minima**	nessuna azione
Se coda > **soglia massima**	tutti i nuovi pacchetti scartati
Altrimenti	scarto con probabilità proporzionale alla lunghezza coda

La politica preventiva ha successo per evitare la congestione

SERVIZI INTERNET e NUOVI REQUISITI

Specifiche differenziate di servizio
più o meno stringenti

best-effort adatto per servizi elastici come i servizi Internet

nessun throughput garantito, ritardi qualunque, non controllo duplicazioni o garanzie di ordine azioni

controlled load simili a best-effort con basso carico ma con limiti superiori al ritardo (con eventuali sforamenti)

servizi elastici e real-time tolleranti

guaranteed load limite stretto al ritardo e massima garanzia

servizi real-time non tolleranti

SERVIZI e NUOVI REQUISITI

IP ⇨ best-effort
TCP ⇨ elastico garanzie di ordinamento, unicità, controllo flusso
OSI ⇨ QoS ottenuto ad ogni livello
Naturalmente, le garanzie di qualità di servizio hanno un costo

Internet in transizione da infrastruttura a basso costo e basse prestazioni a infrastruttura a costi differenziati e prestazioni corrispondenti

Servizi Integrati lavorando a livello di singolo flusso (RFC2210)

Servizi Differenziati aggregando e classificando flussi per diverse qualità (RFC 2475)

http://www.rfc-editor.org/

NUOVI PROTOCOLLI

Evoluzione dei protocolli ⇨ Servizi Integrati

Nuovi protocolli per adeguare Internet in modo da ottenere un maggior controllo delle operazioni e delle risorse compatibilmente con le proprietà best-effort di IP

In generale, si ragiona per flusso e per hop senza considerare troppo la scalabilità

RSVP ⇨ Resource Reservation Setup Protocol

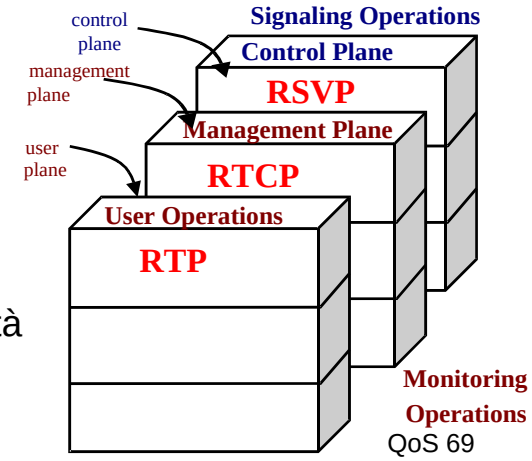
(RFC 2205) protocollo di **segnalazione** per richiedere risorse sui nodi intermedi

RTCP ⇨ Real-Time Control Protocol

gestione dinamica e controllo per mantenere QoS negoziato

RTP ⇨ Real-Time Protocol

(RFC 1889) messaggi generali di operatività con invio affidabile di frame attraverso datagrammi UDP



Servizi Integrati per QoS

Servizi Integrati INTSERV (RFC2210)

Supporto al QoS a livello applicativo con la **distinzione dei flussi**

L'idea dei **servizi integrati** è quella di arrivare a **definire** e **mantenere** un certo **livello di servizio** per ogni **specifico flusso** in un certo dominio di amministrazione o anche in uno scenario globale, sia best effort, sia con QoS **lavorando a livello applicativo**

Una applicazione richiede un certo **livello di servizio (SLA)** specificato usando una **interfaccia** opportuna e un **protocollo di management per un flusso richiesto**

Il protocollo permette di verificare che il servizio si possa fornire (controllo di **ammissibilità**) e di organizzare tutto per fornirlo

La **suite** (come insieme di tre protocolli) non si occupa direttamente delle **azioni locali** e della **garanzia di rispetto del SLA** che deve essere ottenuta a basso livello in modo opportuno (non livello controllo)

*del protocollo locale si devono occupare i **livelli bassi** (di rete) in INTSERV*

Servizi Integrati per QoS

Integrated Services o IntServices - Principio di base

Nella erogazione dei diversi flussi, si deve cambiare il punto di vista

I ***flussi sono considerati uno per uno*** **(con SLA)**

Per ogni flusso, si devono considerare **non solo gli endpoint** ma anche **tutto il cammino** che permette il passaggio e attua il canale fornendo risorse e attivarsi

Il cammino diventa un **cammino attivo e si lavora hop-by-hop**

In genere il servizio prevede

un iniziatore attivo (ricevente o cliente) e

un fornitore del servizio (provider)

che devono poi essere collegati dal cammino attivo più adatto alla fornitura del servizio stesso attraverso intermediari selezionati

RSVP - Reservation Protocol

RSVP Reservation Protocol

Il protocollo specifica come comunicare tra nodi vicini per arrivare a riservare le risorse necessarie a garantire un certo livello di servizio (in modo del tutto separato dal traffico corrente sui canali)

Il **R**e**S**er**V**ation **P**rotocol provvede alla gestione attraverso **informazioni di traffico desiderato** che sono inviate dal mittente al ricevente (***nella direzione della successiva erogazione del flusso***) trattate su sua iniziativa da tutti i **nodi del cammino attivo** per ottenere il servizio stesso e dal **ricevente** di un servizio (***in direzione dal ricevente al mittente***)

Protocollo prima della erogazione e fuori banda (non assieme ai dati utente)

Messaggi scambiati: Path, Resv, ResvTear, PathTear, ResvErr, PathErr, ...

Negoziazione delle FlowSpec (Specifiche di Flusso)

- **TSpec** (descrizione del traffico) inviate sulla rete dal ricevente
- **AdSpec** (opzionale) il mittente conferma la reservation al ricevente

RSVP riserva le risorse in modo unidirezionale (mittente – ricevente)

RSVP - Reservation Protocol

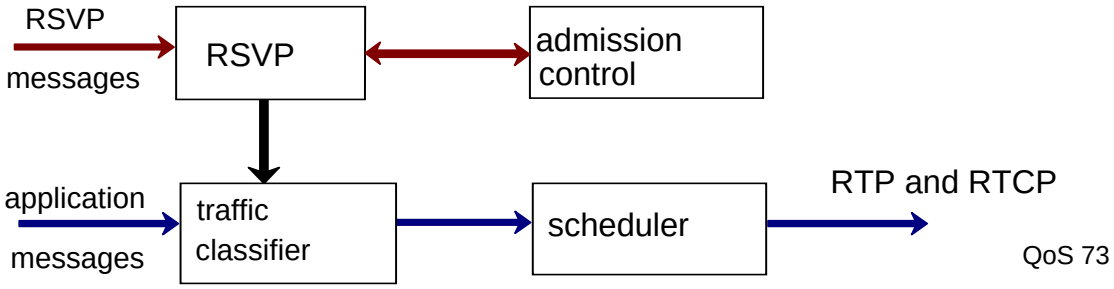
RSVP (RFC 2205) parte di INTSERV (stato soft e due passi)

RSVP **Protocollo a due fasi**, con **soft state,** in cui il **ricevente** di un servizio cerca di **prenotare le risorse** di cui ha bisogno per la durata del servizio stesso

in modo indipendente da eventuali multicast o unicast di routing

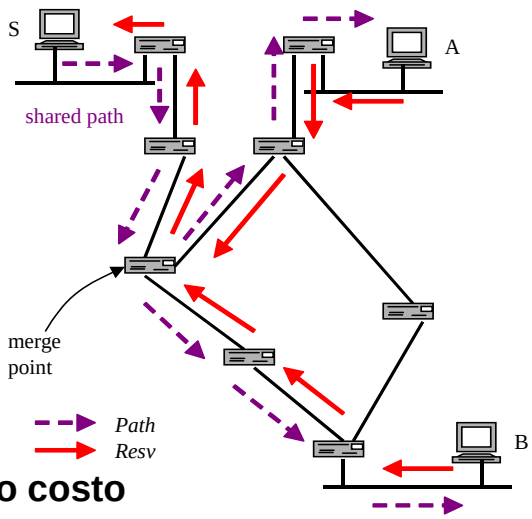
in modo non permanente ma per un intervallo (da rinfrescare)
soft-state

*Si possono riservare risorse in modo **condiviso (tra flussi)** o **fissato** (con possibili ottimizzazioni per la condivisione)*



RSVP - Message Protocol

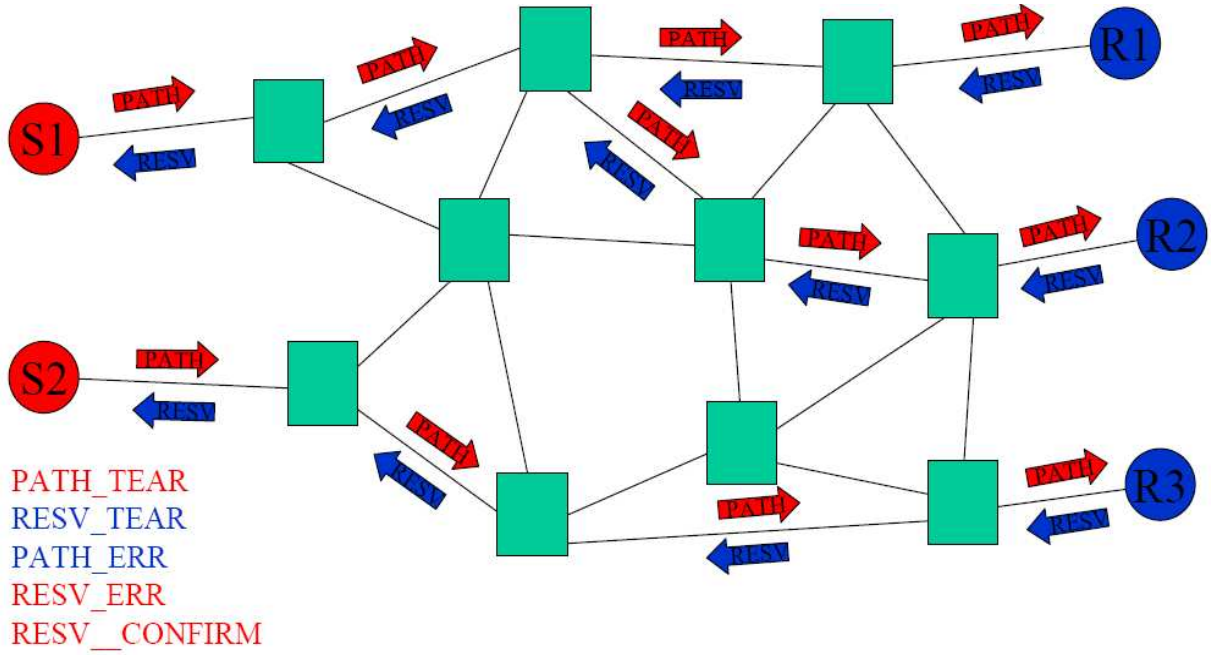
RSVP prevede un protocollo a due fasi con messaggi di **Path** e **Resv**
messaggi **Path** arrivano dai server in **broadcast**
sender: messaggio **Path**
e i riceventi inviano **Resv** per definire cammini
receiver: messaggio **Resv** -
TSpec (+ Rspec e anche in **broadcast**)
refresh del soft-state usando altri **Path** e **Resv**
Si può rispondere con *PathTear* o time-out
sender: **PathTear**
receiver: **ResvTear**



- Uso di **broadcast** dove necessario con **alto costo**
- I nodi mantengono il **soft-state** fino alla prossima reservation
- I cammini e le risorse sono prenotate in modo **privato o condiviso (shared)**

Propagazione Reservation Protocol

RSVP: la prima fase di PATH si propaga dal sender e la seconda contemporanea coi messaggi di RESV in senso opposto



RSVP - Reservation Protocol

RSVP introduce l'idea di lasciare la responsabilità di **riservare risorse al livello di applicazione prima della erogazione (provisioning)**

*Per il **protocollo a due passi** una reservation può bloccarne un'altra producendo **ResvErr***
*Lo stato deve essere mantenuto per **ogni ricevente** e si produce **traffico** per ogni rinfresco dello stato*
*si possono condividere **risorse in multicast** e si devono fornire livelli di servizio compatibili per **riceventi diversi***

Eventi da considerare e riconsiderare

In caso di router failure, la **QoS** può anche degradare fino a best-effort → in questo caso è necessario rinegoziare **QoS durante il provisioning**

Applicazioni e router devono sapere che si usa RSVP e si possono riscontrare problemi con applicazioni legacy
Al momento viene raccomandato solo per *reti locali ristrette* e *non per ambienti globali*

SOMMARIO RSVP

RSVP: Protocollo **single-hop** nella proposta **INTSERV**
(da un nodo a un vicino potenziale) sul cammino attivo

- RSVP ha l'obiettivo solo di **segnalare le informazioni** per poi riservare le risorse necessarie alla QoS
- RSVP è orientato alla operatività **su iniziativa del ricevente**
- RSVP produce **stato su tutti i nodi del percorso** che si viene a stabilire dal mittente al ricevente nella seconda fase
- RSVP prevede uno **stato non permanente** del cammino attivo
- RSVP può consentire anche la **condivisione di cammini attivi**
- RSVP può funzionare sia con **protocolli di routing qualunque,** sia unicast sia multicast durante la loro operatività
- RSVP non è un protocollo di routing ma deve essere compatibile con questi (IPv4 e IPv6 ad esempio)

ALTRI PROTOCOLLI INTSERV

Protocolli di supporto a QoS a livello applicazione (in banda) durante la erogazione (RFC1889)

Considerando come **trasporto il protocollo UDP** (si esclude TCP(?)), si definiscono ed usano due **protocolli a livello di singolo flusso per i dati durante e per la erogazione** (**protocolli single-hop**)

RTP → **R**eal-time **T**ransport **P**rotocol *porte UDP pari*

RTCP → **R**eal-time **T**ransport **C**ontrol **P**rotocol *porte UDP dispari*

che rendono possibile un **controllo della QoS** durante la **erogazione del servizio stesso** rendendo disponibili alcuni indicatori a livello di applicazione (*ovviamente senza garantire QoS ma fornendo informazioni*)

attraverso una **accresciuta visibilità a livello applicativo**

Per la erogazione del flusso, e per tutta la durata

RTP messaggi di marking del traffico, del tempo e applicativi

I messaggi RTP mandati **in banda con numeri progressivi e tempi di passaggio insieme ai dati del flusso**

RTCP messaggi di gestione della connessione

RTP per QoS

Protocolli di supporto al QoS a livello applicativo

- I **flussi di informazione** sono mandati dal sender al receiver attraverso una connessione **applicativa** gestita hop-by-hop
- I singoli pacchetti (**frame del flusso**) sono identificati con tag **numerati successivamente** e possono anche essere **riconosciuti dai classificatori dei diversi router**
- Si possono fornire indicazioni di **tempo di passaggio** per i diversi hop del cammino tra mittente e ricevente
- In caso di pacchetti mancanti, si suggerisce **non** una ritrasmissione, ma una **interpolazione** dei precedenti
- Si prevedono anche **formati differenziati** nella parte **dati dei pacchetti** per andare incontro alle esigenze delle diverse applicazioni

RTP - Real-time Transport Protocol

Real-time Transport Protocol **(dal sender al receiver)**

Ruolo attivo sia per il **sorgente** sia per **mescolatori** (mixer) che possono incidere sul protocollo inserendo tracce del passaggio (***direzione sender – receiver***)

Gli intermediari possono intervenire sul messaggio con timestamp, per aggiungere informazioni per consentire di monitorare la SLA

RTP

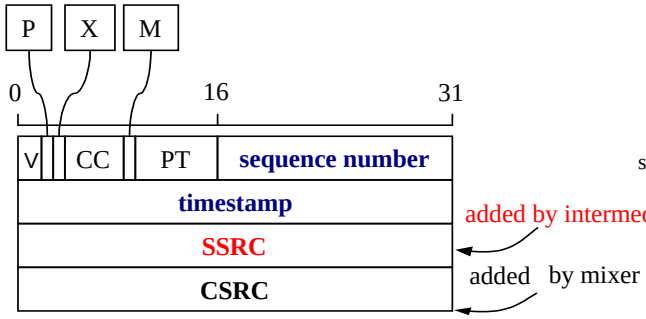
I nodi intermedi (come sorgenti addizionali) possono inserire **informazioni sui messaggi applicativi** che servono per **qualificare ulteriormente** la consegna di informazioni e propagare notizia di **eventuali ritardi**

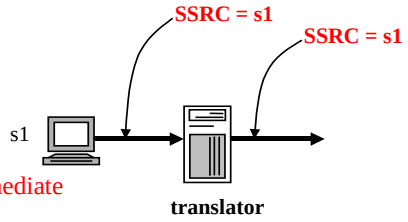
Il **cammino attivo** diventa un **insieme di sorgenti** per ogni nodo di passaggio

Si possono anche considerare **cammini condivisi** che prevedono quindi grafi più complessi con nodi di congiunzione (mixer)

RTP - Real-Time Transport Protocol

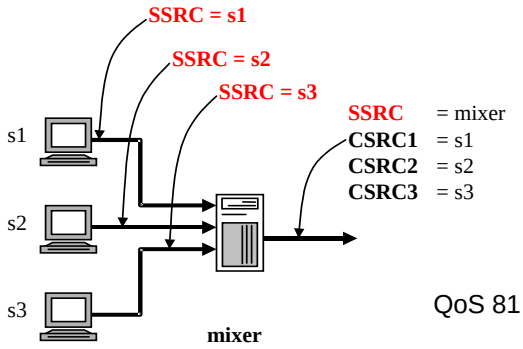
Real-Time Protocol - più sorgenti, primaria, di passaggio nel path (sync. source) e di condivisione (contrib. source)





V	2-bit, version number (=2)
P	1-bit, padding
X	1-bit, signal an extension di header
CC	4-bit, numero di CSRC (CSRC count)
M	1-bit, marker specific per profile
PT	7-bits, payload type, specific of profile
SSRC	synchronisation source
CSRC	contributing source

timestamping in units defined by profile/flow



REAL-TIME TRASPORT CONTROL PROTOCOL

Real-time Trasport Control Protocol RTCP (bidirezionale)

fornisce informazioni globali e sintetiche di **controllo sul flusso** dei dati applicativi con *informazioni sintetiche sullo svolgimento della erogazione e fuori banda*

attraverso messaggi di controllo inviati insieme al traffico (in banda, anzi in contesa con il traffico)

I messaggi di RTCP *viaggiano nelle due direzioni* e permettono di propagare informazioni a tutti i partecipanti, nei due versi, sia relativi alla normale operatività sia ad eventi eccezionali

Obiettivo è propagare 'velocemente' la conoscenza della situazione in atto e dare spazio ad interventi

QoS per flusso

informazioni sui pacchetti: *perdite, ritardi, jitter*

informazioni di end system: *utente*

informazioni di applicazione: *specifiche di flusso applicativo*

Real-time Transport Control Protocol

RTCP Protocollo (associato a RTP) per la gestione dei flussi con QoS e trasporta solo informazioni di controllo del flusso corrente per RTP

per fornire informazioni sintetiche sui parametri dei flussi, tipo ritardo, banda, jitter, ecc.

Mentre i flussi sono in erogazione ed in atto

Obiettivo: **eventuale correzione**

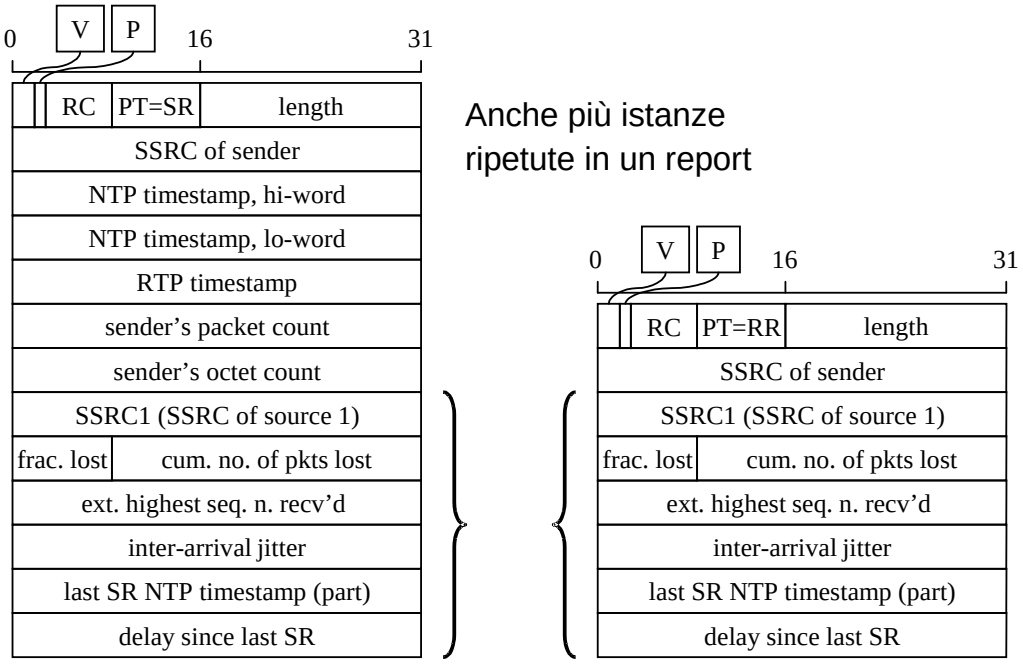
uso di messaggi tipati

RR / SR	Receiver / Sender Report
SDES	Source Description
BYE	Abort di sessione
APP	Specifica di applicazione

Il protocollo RTCP è vincolato ad operare usando le **stesse risorse** (banda) rispetto a **RTP** e viene contenuto in **intrusione**, limitandone l'impegno **percentuale di banda** (5% - 10% di RTP)

RTCP

Messaggi di tipo **RR** e **SR** (**R**eceiver / **S**ender **R**eport)



RTPC - Real-time Transport Control Protocol

Messaggi di tipo SDES descrizione logica del flusso

Source **DES**cription come stringhe ASCII con informazioni del tipo

- 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

RTCP

Messaggi di tipo BYE

BYE specifica l'abbandono una sessione RTP

Un **SSRC** (o SSRC e lista CSRC se mixer) manda questo messaggio, …

fornendo un suggerimento sulle ragioni dell'abbandono

Messaggi liberi di tipo APP

APPlication permette di passare pacchetti application-specific

Un **SSRC** specifica ASCII stringhe 'for name of element' come dati application dependent

In sintesi… per INTSERV

Un flusso applicativo, (fase statica) prima della erogazione

- **provvede al cammino e a riservare risorse con RSVP**
- **poi durante il provisioning viene associato a RTP (e RTCP)**
- **in caso di problemi, anche nuova negoziazione del path, anche localmente agli intermedi (via RSVP)**

RTSP - Real-Time Streaming Protocol

Protocolli Streaming Real Time Streaming Protocol (RFC 2326)

integrazione di uno **streaming Web-based** e trasportato fino al cliente (**RealPlayer**)

Si parte, dopo avere scaricato la *specifica* del file dal server

Il player contatta il server via **UDP o TCP** cercando di ottenere il migliore adattamento sfruttando il solo **buffering locale**

Il ricevente *non aspetta di avere scaricato l'intero brano* (tutti i frame), ma mantiene un *buffer di riproduzione* in cui siano presenti almeno alcuni frame

- se UDP, aspetta 2-5 secondi e poi comincia a mostrare
- se TCP, si usa un buffer più ampio

Politiche a pull e push sul server con tecniche di watermark per la sincronizzazione (se sotto soglia, si comincia a chiedere in pull)

Si usano **tecniche di interleaving** per ovviare a perdita di pacchetti

DIFFSERV (Servizi Differenziati)

Servizi Differenziati (DIFFSERV RFC 2474, 2475, …)

L'idea è di differenziare i **servizi offerti in classi diverse** con caratteristiche di **maggiore scalabilità** e supportando la differenziazione a **basso livello, ossia a livello di rete OSI**

I servizi differenziati sono lasciati ad un dominio specifico di applicazione e un gruppo di IETF sta definendone diversi

I servizi sono usufruiti a livello utente e di comunità di utenti, senza **troppo coinvolgimento dell'utente** e con **utilizzo più facile** degli INTSERV ed adatti per applicazioni legacy

I pacchetti sono marcati a **livello di rete** (non a livello applicativo ma di rete) e sono riconosciuti e trattati dai router in modo **aggregato e diretto**

NON si lavora per ogni flusso di informazioni, ma

aggregando classi di flussi a livello rete

DIFFSERV (Servizi Differenziati)

Si definiscono e si usano **classi di servizio diverse**: ad esempio

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

e anche

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

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

Service Level Agreement (SLA) basato sulla classificazione

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

Un flusso viene classificato e poi si va in automatico, inserendolo nella sua classe

DIFFSERV (Servizi Differenziati)

Classi di servizio RFC3246 **expedited forwarding**

Expedited forwarding vs. Regular

I router devono mantenere almeno **due code differenziate** e garantire la consegna dei **pacchetti expedited** in ogni hop (Per-Hop Behaviour)

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

Si crea una connessione punto a punto tipo linea condivisa tra endpoint

Service Level Agreement (SLA) (tipo 80 –20)

I pacchetti devono ricevere almeno un **Weighted Fair Queuing**

Classi di servizio RFC2579 **assured forwarding**

Quattro **classi di priorità** con **tre livelli di trattamento** in caso di **congestione** (basso, medio, alto)

I diversi pacchetti devono essere marcati a trattati in modo differenziato

Meccanismi per Servizi Differenziati

DIFFSERV possono usare molti meccanismi diversi per differenziare servizi

il più praticabile sembra essere il byte detto **DS (Differentiated Service)** nell'header di ogni pacchetto (Type of Service - Service type, o **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 + altri **campi**
- aggregazioni di behavior (BA): solo **DS**

DS codepoint dipendenti dallo scenario di applicazione

Si tentano per-hop behaviour (PHB) aggregando flussi nella rete gestita

Un flusso viene classificato all'ingresso e messo nella coda corretta

ESTENSIONI per QoS (RFC1889)

Necessità di 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, Differentiated Service) per condizionare / ricondizionare il traffico

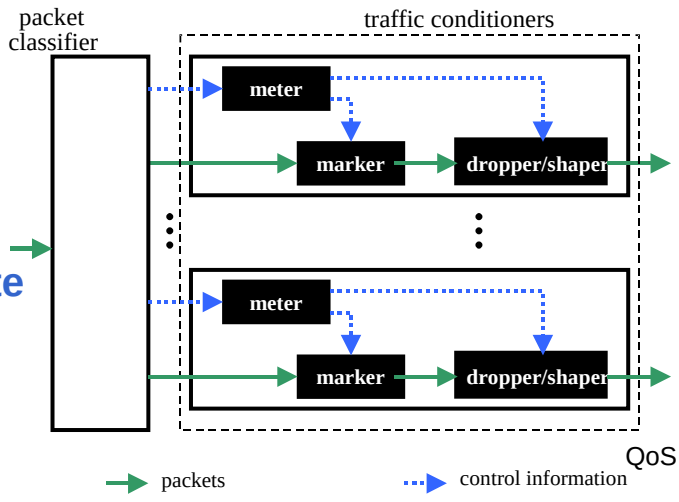
Possibilità di avere

Shaper

Dropper

sui pacchetti

e intervenire attivamente sul traffico in atto



DIFFSERV MULTIFIELD

I classificatori di traffico lavorano nella selezione dei pacchetti sulla base delle **informazioni contenute negli header**, nel modo più ampio possibile (*tenendo conto di ogni informazioni possibile*)

Si possono considerare

- le **porte estreme**,
- 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** e sono sperimentati ancora per zone limitate

Spesso uso **congiunto dei due approcci insieme**

in aree distinte ma integrate (IntServ e Diffserv)

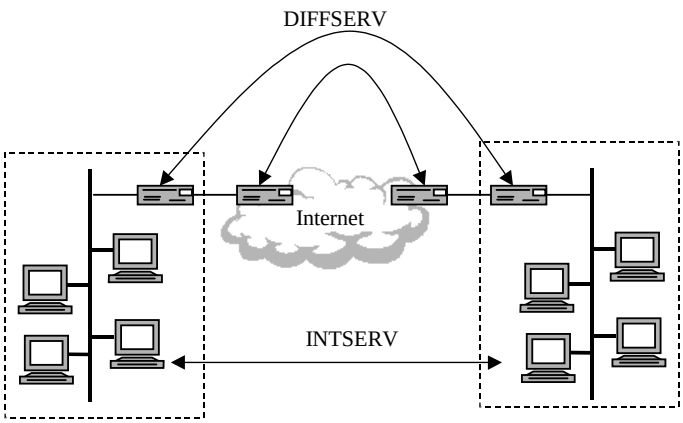
Nuove Proposte DIFFSERV

INTSERV e DIFFSERV insieme

Al momento sono in fase di sviluppo sia i protocolli di **tipo differenziato**, sia di **tipo integrato**

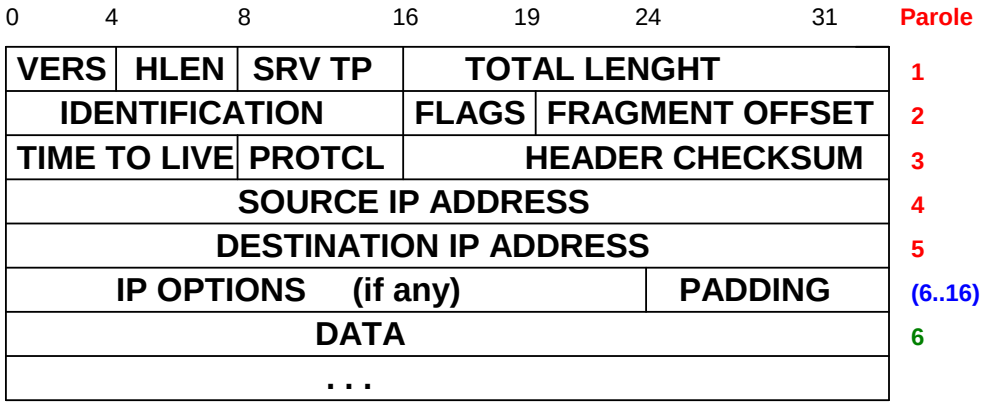
anche se i **servizi differenziati** sembrano essere **più scalabili e fornire prestazioni anche a servizi legacy**

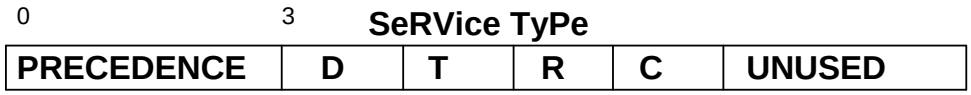
Naturalmente, i router devono fornire i nuovi servizi



HEADER – IPv4

Header IP tradizionale





Uso dei 7 bit insieme, per identificare i flussi e le classi

IPv6

Internet Protocol v6 (IPv6)

nuove proposte di sistemi di routing e di nomi a fronte dell'esaurimento degli indirizzi IP

*solo **2,11 M reti** (alcune classi C libere), 3,72 G connessioni*

IPv6 => 128 bit / 16 byte forte estensione del sistema

mantenendo anche compatibilità con IPv4 (7 10^{23} indirizzi per metro^2)

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 con obiettivi diversi:

Facilitare **multicast**, **roaming**, **sicurezza**, **QoS**, …
limitando tabelle di routing e rendendo il routing più efficiente

Permettere **evoluzioni e coesistenza**

INDIRIZZI IPv6

Internet Protocol v6 (IPv6)

Gerarchia di indirizzi divisi per **forniture di servizio** e **indirizzi geografici**, e anche per **usi locali** e **non visibili**

Inoltre, si riconoscono funzionalità per:

- **point-to-point** *(anche per usi speciali: nascosti, compatibili IPv4, …)*
- **multicast** *(con scope)*
- **anycast** (*il più vicino o più comodo di un insieme di destinatari*)

Non sono previsti broadcast (ma solo multicast)

Con attribuzioni parziali

0: *IPv4*

1: *OSI*

2: *Novell*

…

255: *Multicast*

DATAGRAMMI IPv6

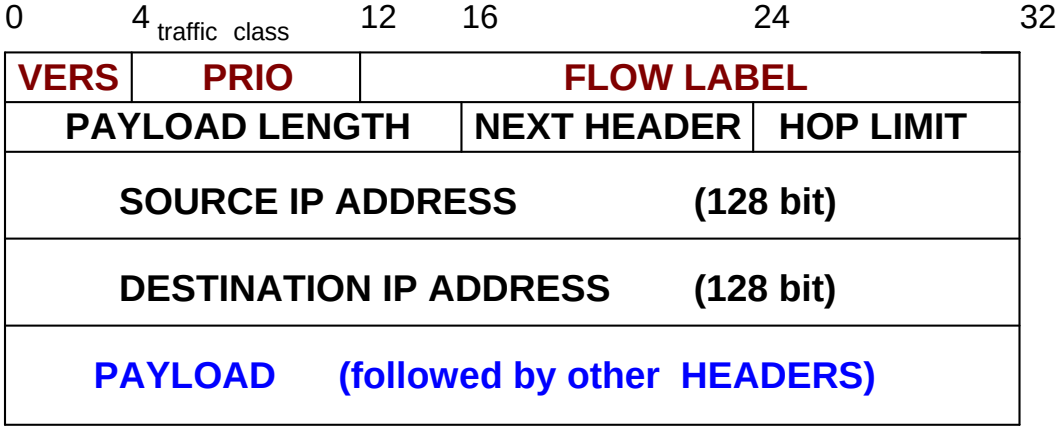
Internet Protocol v6 (IPv6)

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 **eventuali header di estensione**

Nessuna **frammentazione** e **checksum**



HEADER – IP VERS 6

PRIO Type of Service (ToS) 0-3 best effort 4-11 Streaming e QoS

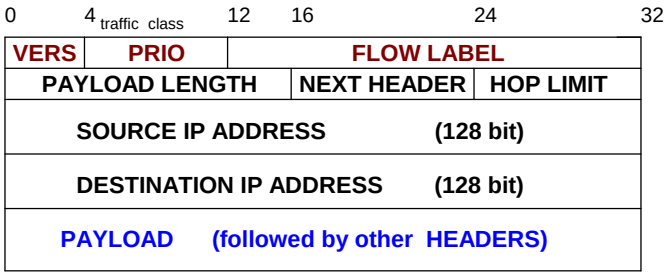
FLOW LABEL 20 bit tiene traccia di flussi nei diversi cammini

PAYLOAD lunghezza minima 536, massima 64K

NEXT HEADER (type length value)

uso di estensioni segnalati con header aggiunti

hop by hop (jumbo)
routing (hop by hop)
fragment
authentication
encapsulating security payload
destination options



HOP LIMIT come il time to live in hop (IPv4)