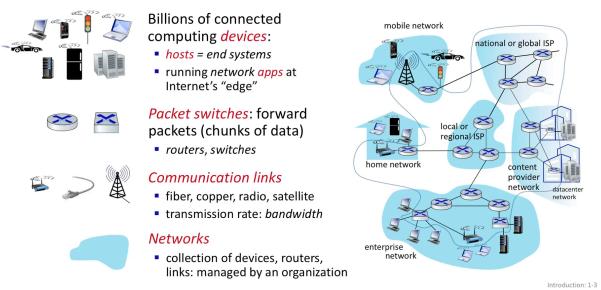


RETI

L01 >

- > Definisci formalmente cos'è una rete.
- > Definisci formalmente cos'è un device/host.
- > Definisci formalmente cos'è un communication link.
- > Definisci formalmente cosa sono router e switches.
- > Definisci formalmente cos'è internet
- > È una rete?
- > Definisci l'altro modo in cui si puo' definire internet.
- > Definisci formalmente cos'è un protocollo.
- > Da cosa e' composto?
- > Cosa succede se si comunica senza un protocollo?
- > Descrivi le categorie di reti possibili.
- > Descrivi cosa sono e come funzionano le reti via cavo.
 - > Quali sono le velocita' in up/down?
 - > Quali sono le particolarita' rispetto alle altre tipologie di reti?
- > Descrivi cosa sono e come funzionano le DSL.
 - > Quali sono le velocita' in up/down?
- > Descrivi i vari tipi di reti LAN:
 - > Quali sono le loro velocita' in up/down?

The Internet

• INTERNET:

È la connessione fra reti indipendenti diverse tramite delle reti messe a disposizione dagli ISP per scambiare traffico fra sorgenti e destinazioni diverse.

• RETE: È un collegamento fra dispositivi diversi.• DEVICE: Detto host, è un dispositivo finale

ES: PC, Tablet, server ecc...

• SWITCHES.

RW / X SW o ROUTERS

Dispositivi che hanno il compito di fare il forward dei pacchetti di dati scambiati.

• COMMUNICATION LINKS

Definiscono dei metodi di comunicazione fra devices e switches/routers.

ES: Fibra ottica, Cavi di rame, Satellite, Radio ecc.

Internet può essere anche descritto come un'infrastruttura che fornisce servizi alle applicazioni, le quali usano l'interfaccia socket per consentire ai programmi di comunicare tramite internet.

PROTOCOLLO >

In rete rappresenta un'insieme di regole per scambiare messaggi in rete.

Es: A Livello fisico, il protocollo di scambio è in BIN.

Il protocollo definisce

- formato e l'ordine dei messaggi scambiati
- azioni intraprese quando si invia / riceve il messaggio.

Il non rispetto del protocollo implica la perdita del pacchetto.

STRUTTURA GENERALE RETE >

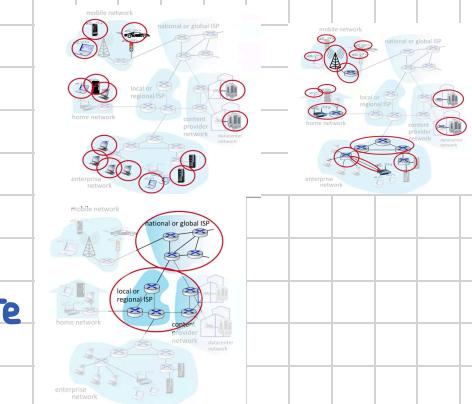
- NETWORK EDGE: Comprendono gli host

- ACCESS NETWORK:

Collega il NET. EDGE al resto di internet tramite edge router. (1 router in uscita).

- NETWORK CORE

Comprendono gli ISP sia nazionali che globali che hanno il compito di collegare reti diverse anche usando altri ISP per il collegamento.



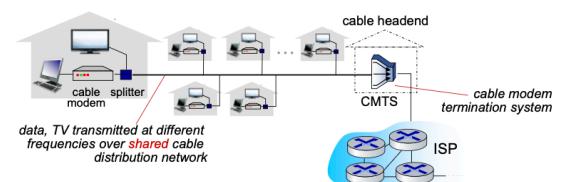
ACCESS NETWORKS > TIPOLOGIE DI RETE

- VIA CAVO TU

Rete che usa le infra esistenti della TV per accedere ad internet

- FUNZIONAMENTO

Un cavo HFC (Hybrid Fiber Coax) si divide per distribuire il segnale TV e internet. Il cavo arriva ad uno splitter che divide i segnali TV / internet.



- CABLE MODEM: Modem usato per l'accesso a internet

che usa freq. Divided Multiplexing

Lato net. core, si usano i CIRTS come splitter.

- VELOCITÀ: Asimmetrica

→ Down: 40 Mb/s - 1.2 Gb/s → Up : 30-100 Mb/s

ND: L' HFC è un canale condiviso. Per cui in downstream la velocità dimin. se ci sono troppi downloads. In upstream invece ci possono essere **collisioni**

- DIGITAL SUBSCRIBER LINE (DSL)

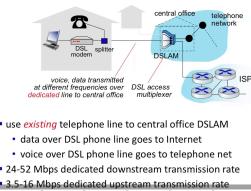
Rete che usa le infra telefoniche per l'accesso.

Usa un comm. link che usando FDM trasmette

- Downstream, Upstream di rete
- Segnale telefonico

Che alla stazione centrale verranno divisi tramite un DSLAM multiplexer verso le rispettive reti.

Access networks: digital subscriber line (DSL)



- VELOCITÀ: Asimmetrica

→ Down: 24-52 Mb/s → Up : 3.5 - 16 Mb/s

- LAN: RETI AZIENDALI / RESIDENZIALI

Sono reti locali che possono collegare i dispositivi tramite:

- ETHERNET: Collegamento tramite cavo intrecciato

→ Vel. Host: 100 Mb/s - 10 Gb/s

- WIFI: Collegamento senza fili basato su Access Points

→ Vel. 450 Mb/s

- CELLULAR: Connessioni cellulari geografiche ad alta copertura.

→ Vel: 10's Mb/s

L02 >

- > Cosa sono le reti a pacchetti e a circuiti?
- > Rete a pacchetti: Come viene calcolato il delay? Come vengono suddivisi i pacchetti?
- > Descrivi tutti i tipi di communication links
- > Network Core:
 - > Cos'è?
 - > Quali sono le sue funzioni principali?
 - > Cos'è lo store-and-forward? Come viene integrato nel network core?
- > Circuit switching:
 - > Per quanto tempo vengono assegnate le risorse?
 - > Che approccio usa nella gestione delle risorse? Quali sono i vantaggi e svantaggi?
 - > Come vengono **divise** le risorse? Indica vantaggi e svantaggi
- > Network Core 2:
 - > Perché è nato?
 - > Quali sono le tipologie principali di attori nelle reti nel net. core? Come interagiscono fra di loro?
- > Packet delay: Quali sono le componenti del ritardo in generale?

Gli host possono inviare dati con 2 metodologie:

- Tramite rete telefonica → rete basata a circuiti
- // rete internet → rete basata a pacchetti

Soltamente si usa la seconda

RETE A PACCHETTI >

La rete internet usa i pacchetti, ovvero segmenti di bit della stessa dimensione (tranne l'ultimo) in cui è racchiusa l'informazione.

INVIO E DELAY DI INVIO

Il delay iniziale della trasmissione dei pacchetti:

$$\text{Packet} = \frac{L \text{ (bits)}}{\text{Transmission Delay R (bits/s)}}$$

Dove:

- L: è la dimensione in bit di ogni pacchetto
- R: È la bandwidth, ovvero la banda di trasmissione del comm. link.

NB: I bit hanno base **decimale**

ES: Bandwidth: 10 kbit/s , L= 30 kbit

$$\text{PTD} = \frac{30 \text{ kb}}{10 \text{ kb/s}} = 3 \text{ s}$$

COMMUNICATION LINKS >

Le tipologie possibili sono:

CANI DI RETE:

Cavi che sono più o meno protetti con schermature e dotati di porte standard (RJ41, RJ45)

CANI COASSIALI:

Cavi usati per le trasmissioni TV (cable modems)

con più canali di comunicazione.

- CANI DI FIBRA OTT.

Cani di fibra in vetro che trasmettono segnali di luce con bassa latenza e interruzioni.

- WIRELESS RADIO

Trasmissione senza fili : Wi-Fi, 4G/5G, Bluetooth, satellite, micro-onde, NFC ...

Con tipologie di trasmissione pari a:

- Half-duplex: Consente di inviare / ricevere in momenti diversi

- Full-duplex:    nello stesso momento

NETWORK CORE 

Il network core assume una duplice funzione :

1. ROUTING (Globale)

La pianificazione di un percorso in maniera ottimale.
È effettuata come scelta globale prima dell'invio
del pacchetto (a livello di topologia)

Questa viene aggiornata in continuazione e calcolata
tramite algoritmi (ES: Dijkstra)

2. FORWARDING (Locale)

La scelta del router di destinazione durante
l'invio del pacchetto nella rete.

Questa viene presa dal router corrente in locale
per

- Scalabilità : Sistema centralizzato → Intasato.
- Efficienza : Aspettare sistemi terzi rallenta.
- PROCESSO DECISIONALE

Ogni pacchetto in arrivo avrà un indirizzo di destinazione nell'Header.

Il router farà un confronto nella sua forwarding table e deciderà in base all'indirizzo.

• STORE & FORWARD

Ogni router aspetta la ricezione dell'intero pacchetto dall'Host prima di effettuare il forward.

Questo perché senno

- Non avrebbe abbastanza info per capire dove andare
- == la certezza di mandare senza errori.

Lo store richiede:

- memoria per la coda di pacchetti da inviare
- delay aggiuntivo → PTD fra source e router

La coda si formerà quando arrivano più msg di quanti se ne possano gestire. Coda piena → Scarto.

CIRCUIT SWITCHING >

Comune nelle reti telefoniche, prevede di mantenere le risorse allocate durante tutta la sessione di comunicazione.

Ogni risorsa non è condivisibile, che garantisce performance ma comporta spreco di risorse.

• DIVISIONE RISORSE

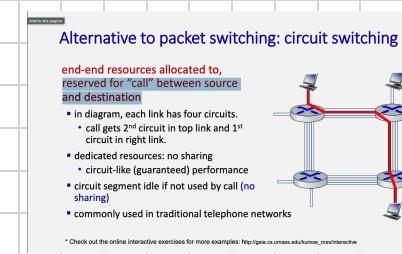
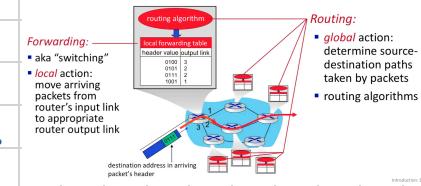
Le risorse possono essere riservate con

• FDM: Frequency Division Multiplexing

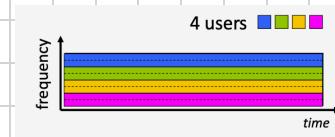
Prevede di suddividere le bande di freq.

fra i dispositivi + NO vincoli tempo - Vel. limitata a freq.

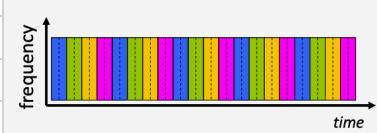
Two key network-core functions



* Check out the online interactive exercises for more examples: <http://gate.cs.unsw.edu/~kumaraswami/interactive>



TDM: Time Division Multiplexing



Prevede di allocare time slots in cui

ogni host potrà comunicare con tutte le freq.

+ Vel. elevata - Nessun altro può comunicare nel tempo

CORE NETWORK > STRUTTURA

Collegare N host fra loro richiederebbe $O(N^2)$ connessioni, cosa non fattibile.

Il core network è nato per semplificare le comunicaz. in modo da creare reti di passaggio.

Elementi chiave sono:

- ISP: Business nati per creare e gestire
Tier 1: AT&T, NTT...
network core intermedi.

- Peering Link: Canali di comm. fra 2 ISP

- Inter. Exchange Point: Ponti di scambio di traffico fra più ISP.

- Content Provider Network:

Network dedicati creati da fornitori di contenuti (es: Google) per bypassare gli ISP.

PACKET DELAY >

Il Packet Delay verrà quindi da

$$d_{tot} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

Dove

• d_{proc} : Tempo impiegato dal Router per proc.

• d_{queue} : // // in coda di attesa di trasm.

• d_{trans} : Tempo di trasmissione del Router del pacchetto

• d_{prop} : // di propagazione o di effettivo viaggio d/s dist/teluso. Di ogni Bit

L03 >

-> **Packet intensity:**

- > Che cos'è?
- > Da cos'è composta?
- > Cosa succede per valori alti? Per valori bassi?

-> **Throughput:**

- > Che cos'è?
- > Come puo' essere misurato?
- > Che cos'è e come si verifica il bottleneck

-> **Network security:**

- > La rete è nata con funzionalità di sicurezza built-in?
- > Quali sono gli attacchi comuni che coinvolgono le reti?
- > Quali sono le difese principali che proteggono le reti?

-> **Layered stack:**

- > Con che architettura viene implementato il sistema di comunicazione in rete?
- > C'è una differenza di implementazione fra i vari livelli di implementazione in rete?
- > Descrivi per ogni livello:
 - > Nome
 - > A cosa serve
 - > Quale unita' viene usata
- > Come vengono usati i livelli nella comunicazione fra due dispositivi?
- > Tutti i dispositivi accedono a tutti i livelli della pila protocollare?

L02: Packet/Circuit Networks & Delay

PACKET QUEUING DELAY \rightarrow RIVISTO

Si definisce come Packet Intensity la seguente formula:

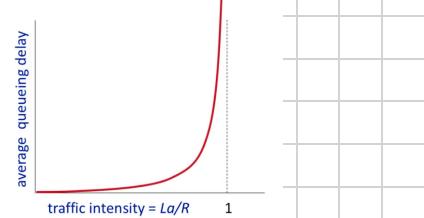
$$\frac{\text{Packet Intensity}}{\text{Intensity}} = \frac{L \cdot a}{R} = \frac{\text{"velocità media di arrivo" bit/s}}{\text{"Banda di servizio media"}}$$

Dove:

- L : È la dim. del pacchetto
- a : È la vel. media di arrivo dei pacchetti in coda
- R : È la bandwidth.

Da cui si può dedurre che :

- Per $La/R \approx 0$: Ci sarà poco delay.
- $\approx \frac{La}{R} \rightarrow 1$: Il delay aumenterà esponenzialmente
- $\approx \frac{La}{R} \geq 1$: Il delay è too



TRACEROUTE >

È un comando che consente di tracciare i pacchetti.

Funziona mandando ripetutamente 3 pacchetti verso ogni router fino alla destinazione misurando i delay tra invio e risposta fino all'host di partenza.

OSSERVAZIONE FUNZIONAMENTO

Per ogni IP, ci saranno i 3 tempi o 't' se il router non ride

```
extremis@MacBook-Pro-di-Marcelo ~ % traceroute www.pornhub.it
traceroute to pornhub.it (34.95.33.229), 64 hops max, 40 byte packets
 1  192.168.1.1 (192.168.1.1)  2.139 ms  0.865 ms  0.534 ms
 2  172.25.0.1 (172.25.0.1)  5.492 ms  3.982 ms  3.444 ms
 3  * * *
 4  * * *
 5  83.224.46.238 (83.224.46.238)  9.057 ms
    185.210.48.39 (185.210.48.39)  6.779 ms
    83.224.46.238 (83.224.46.238)  6.213 ms
 6  185.210.48.3 (185.210.48.3)  6.103 ms
    83.224.46.237 (83.224.46.237)  6.154 ms
    185.210.48.39 (185.210.48.39)  14.305 ms
 7  229.33.95.34.bc.googleusercontent.com (34.95.33.229)  101.791 ms
    83.224.46.237 (83.224.46.237)  7.169 ms  7.811 ms
```

THROUGHPUT >

È il tasso con cui determinati bits/s verranno inviati.

Può essere

- ISTANTANEO : Misura istantanea con cui il destinatario sta ricevendo i dati
- MEDIO : Misura media di ricezione su periodi più lunghi
Dim \rightarrow Pacchetto
Tempo \rightarrow Di ricezione

ci possono essere conseguenze sulle velocità:

• BOTTLENECK LINK:

Regola per cui prevale il minor throughput in una comunicazione. Solitamente il minor throughput si trova nel Network Edge.

Es: Date:

- R: connett. "Core" da 10 hosts
- R_c, R_s : Connessioni Network edge

Il throughput di R è $R/10$ per condividerlo con ogni host possibile. Sarà quindi min($R_c, R_s, R/10$)

NETWORK SECURITY >

Il design della rete internet non ha tenuto conto la possibilità che non tutti gli utenti siano 'good actors'.

• ATTACCI COMUNI

Attacchi comuni dei 'Bad Actors' sono:

- PACKET SNIFFING: Wireshark

L'attaccante si intromette fra l'host e il router e 'rubai' i pacchetti trasmessi.

- IP SPOOFING:

L'attaccante prova ad iniettare pacchetti al router/dest. fingendosi l'host.

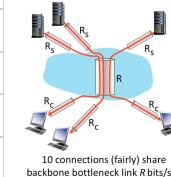
- DENIAL OF SERVICE

L'attaccante blocca il servizio bombardando di traffico osando host penetrati (botnet).

• DIFESA

Possibili linee difensive includono

Throughput: network scenario



- per-connection end-end throughput: $\min(R_c, R_s, R/10)$
- in practice: R_c or R_s is often bottleneck

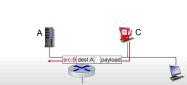
* Check out the online interactive exercises for more examples: <http://cs.uwaterloo.ca/~khoudeim/exercises.html>

packet "sniffing":

- broadcast media (shared Ethernet, wireless)
- promiscuous network interface reads/records all packets (e.g., including passwords!) passing by



IP spoofing: injection of packet with false source address



Denial of Service (DoS): attackers make resources (server, bandwidth) unavailable to legitimate traffic by overwhelming resource with bogus traffic

- select target
- break into hosts around the network (set botnet)
- send packets to target from compromised hosts



- Authentication: ES: Simcard nella rete cellulare.
- Confidentiality: Tramite la crittografia
- Integrity Checks: Uso di firme digitali o altri modi per rilevare le manomissioni.
- Access Restrictions: Tramite VPNs protette
- Firewall: Elementi intermedi programmabili per mitigare gli attacchi e filtrare i pacchetti.

LAYERED STACK >

I protocolli di rete sono stati pensati a livelli / layer, dove ogni livello usa i protocolli del livello precedente per la sua implementazione.

• DETtagli IMPLEMENTAZIONI

Ogni protocollo può essere implementato via HW, SW o entrambi. Ciò varia in base all' "altezza" del livello.

ES: Liv. fisico \rightarrow HW Liv. Rete \rightarrow HW Liv. App \rightarrow SW

I livelli sono

• APPLICATION: Unit: Message M

Concentrata su applicazioni e protocolli di rete. Protocolli:

- HTTP/S: Per invio / ricezione documenti web.
- FTP: Per trasferimento di file
- DNS: Per traduzione dei domini in IP.

La singola unità scambiata è detta **message**.

• TRANSPORT: M + H_T

Trasporta messaggi dell'app. layer da un processo cellulare.

- TCP: Trasferimento con garanzia di consegna
- UDP: // senza // == ==

Il liv. Transport **incapsula** il message in arrivo, aggiungendo un transport header H_t creando un **segment**. H_t contiene informazioni per consentirne il funzionamento.

- Network: $M + H_t + H_n$

Si occupa di trasferire pacchetti o datagramme fra hosts.

Il trasferimento è **best effort**, ovvero non garantito.

Il liv. network **incapsula** il segment con i suoi header H_n definendo il **datagram**. (ES: l'IP è negli headers)

- L2N : $M + H_t + H_n + H_l$

Si occupa di instradare un datagramma al nodo di rete successivo.

Incapsula il datagram con i suoi header per ottenere il **frame**

- Physical

Trasferisce fisicamente i bit di un nodo al successivo

Quindi per ogni messaggio:

- All'invio: Si scende di liv. e si aggiungono header

- Alla ricezione: Si sale di liv. e si tolgono header.

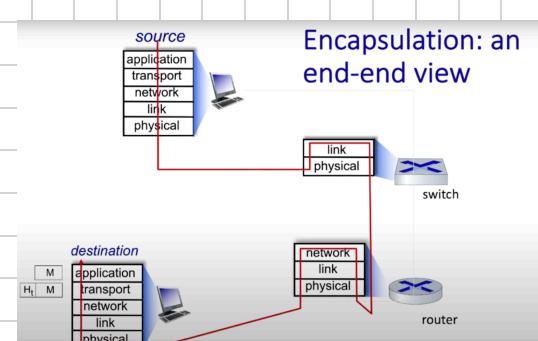
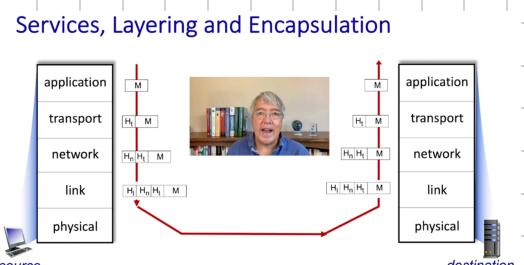
- Giro GENERALE

Generalmente, un host attraverserà

1 + routers/switches che avranno

solo i livelli finali in quanto

necessitano solo di quelli.



L04 >

-> **Socket:**

- > Che cos'è?
- > Da chi/cosa viene usata e per cosa?
- > Chi la usa di cos'altro ha il controllo?
- > Come sono identificabili i socket?
- > Perche' e' necessario quel metodo di identificazione?
- > Per cosa e' pensata esattamente l'identificazione del socket?
- > Prevedono funzionalita' di sicurezza?

-> **Overview Transport Layer:**

- > Descrivi i principali protocolli di trasporto così come le loro caratteristiche principali.
- > Come si integrano con la sicurezza e il throughput?

-> **TLS:**

- > Che cos'è?
- > A che livello del layered stack è applicato?

-> **HTTP:**

- > Quali sono le sue caratteristiche principali?
- > Quali sono le sue well-known-ports?
- > Come possono essere gestite le connessioni in HTTP?
- > Che vantaggi e svantaggi portano gli approcci?

È un'interfaccia SW o API usata dai processi per la comunicazione con altri processi nella rete.

- SCOPE DI GESTIONE:

I progettisti hanno il controllo del livello applicativo e del socket nella creazione di programmi, in quanto avvengono in **user space**. Non hanno controllo nei livelli inferiori, gestiti in **Kernel space**.

- STRUTTURA DEL SOCKET: IND: PORTA

Sono identificati da un indirizzo IP e una porta, le quali vengono usate per identificare i socket dest. nella comunicazione.

NB: Non prevedono cifratura, i dati arrivano così come sono.

TRANSPORT > PROTOCOLLI

A livello transport sono disponibili i seguenti protocolli:

- TCP:

È un protocollo di trasporto affidabile, che si accerta **ordinata** della ricezione **✓ dei pacchetti**. Altre caratteristiche:

- + Flow Control:

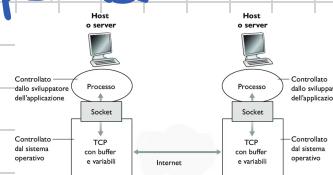
Si sincronizza con la capacità di ricezione del receiver per evitare di mandare troppe richieste.

- + Congestion Control:

Si auto-regola il throughput per evitare congestioni nei nodi intermedi (ES: ↑ 70%, TCP 50% → 30%)

- + Connection Oriented:

Necessita di una connessione. Manderà pacchetti fino a che non viene chiusa.



Il protocollo invece non prevede garanzie di throughput minime o sicurezza.

• UDP

È un protocollo di trasporto non affidabile, ovvero non si accerta dell'arrivo dei messaggi.

TCP > Sicurezza con TLS (Transport Layer Security)

È un layer di cifratura usato per mettere in sicurezza le connessioni TCP.

È implementato a liv. application e applicato **prima** dell'invio al socket. IETF : Ente che crea i protocolli

Comandi Utili > # Non cercano nelle slides

- netstat : Mostra le connessioni attive [-l: in cui è server] listen

- nc : Consente di creare socket [-l [nc -l 123.0.0.1:8000]]

WEB & HTTP >

È un protocollo testuale per l'invio/ricezione di righe.

• CARATTERISTICHE GENERALI

- USA TCP : Solitamente in porta 80 o 443 (con TLS).

- È STATELESS : Non mantiene memoria sulle richieste.

• TIPOLOGIE DI CONNESSIONI HTTP

Le connessioni HTTP a livello di connessione TCP può essere:

- PERSISTENTE:

La stessa connessione TCP viene usata per scambiare più messaggi.

- NON-PERSISTENTE:

Viene creata una nuova connessione per ogni richiesta.

La seconda non conviene perché aumenta la latenza (dicono).

L05 >

-> RTT & HTTP:

- > Che cos'è e a cosa serve RTT?
- > Quanti RTT sono necessari nelle connessioni HTTP? Perché?
- > Nelle connessioni persistenti ci sono piú o meno RTT delle connessioni non persistenti?
- > Interazione con HTTP:
 - > A partire da quale versione HTTP vengono usate le connessioni persistenti?
 - > Che header HTTP viene usato per la gestione delle connessioni persistenti? Con che valori?

-> Cookies:

- > Cosa sono e a cosa servono?
- > Come funziona il processo di utilizzo dei cookies?
- > Quali header HTTP viene usato per gestire i cookies? Con quali valori?
- > Hanno una durata definita?
- > Qual'è la differenza fra first-party cookies and third-party cookies?
 - > Di quale header HTTP si servono i third-party cookies?

-> Cache Web:

- > A quale scopo vengono usate?
- > A che livelli possono venire usate?
- > Header di cache web in HTTP:
 - > Di quale header si servono?
 - > Da chi viene restituito?
 - > Che valori puo' contenere? In quali casi?

COMUNICAZIONI HTTP >

• RTT: Round Trip Time

Rappresenta il tempo impiegato per invio e ricezione di un pacchetto di piccole dimensioni (ES: pacchetto TCP)

La configurazione di una connessione TCP richiederà 2 RTT
 questo significa che nelle comunicazioni non persistenti TCP ci vorranno $2 \text{ RTT} + T_{file}$

• GESTIONE RICHIESTE / RISPOSTE

È necessario 1 RTT di handshake TCP, a partire

da HTTP/1.1 si usano le conn. persistenti usando l'header Connection:V

COOKIES >

I cookies sono nati con l'obiettivo di aggiungere uno stato al protocollo HTTP.

• FUNZIONAMENTO

1. Prima richiesta: Sarà senza cookie

2. Prima risposta:

Il server creerà un cookie (ID di sessione) e lo restituirà al client nel response header Set-Cookie: <cookie>.

3. Richeste successive

Il client salva il cookie e lo manderà nelle richieste successive nel request header 'Cookie'.

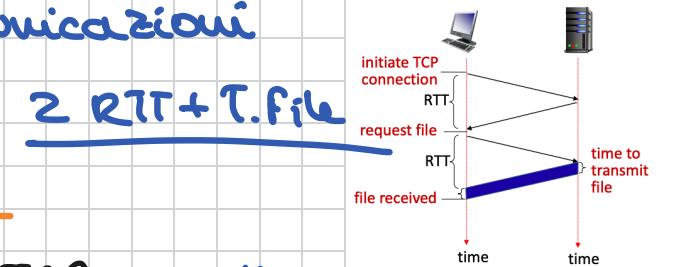
NB: I cookies hanno una durata definita.

Si può fare una distinzione fra i cookies:

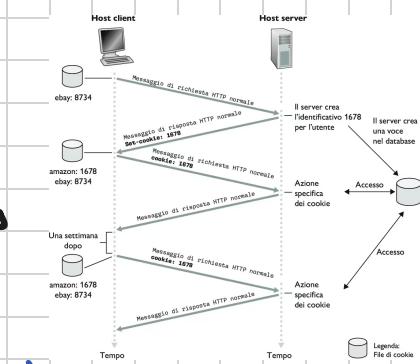
• First-Party Cookies: Da siti che si è scelto di visitare

• Third-Party Cookies:

Da siti che non è scelto di visitare, solitamente ads.



Keep-Alive
Connection:V



• FUNZIONAMENTO THIRD-PARTY

Cookies: tracking a user's browsing behavior

Per vedere gli ad, il browser + sito

lanciano una richiesta aggiuntiva al

sito del provider di ads (es: AdX.com).

La richiesta conterrà il Referrer header, che conterrà informazioni sull'ultima richiesta (documento visitata).

L'Ad provider staccherà quindi un cookie per identificare l'host con la visita al determinato sito.

CACHE WEB >

Le cache web vengono usate per diminuire il tempo di attesa / RTT nelle richieste web - HTTP.

• TIPOLOGIE DI WEB CACHES

Le cache web possono avvenire a livello

- CLIENT: Tramite il local storage del Browser.
- SERVIZIO: Tramite nodi più vicini ai client da servire.

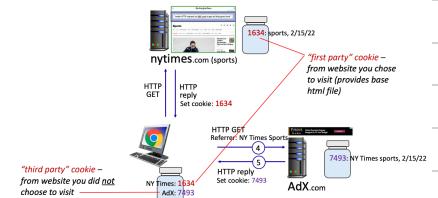
• DURATA

I server impostano l'header Cache-Control per segnalare

le policy di cache da applicare sulla risorsa alle cache:

→ Cache-Control: max-age=<s> : Impostabile in cache per **s** secondi.

→ Cache-Control: no-cache : Cache non presente



L06 >

-> **Proxy web:**

- > Che cos'è?
- > A cosa serve?
- > Come si integra con le cache web.
- > In scenari di performance basse, conviene migliorare il communication link oppure usare un proxy web?
- > Conditional GET:
 - > Che cos'è?
 - > Che HTTP header viene usato per poterla usare?
 - > Si può usare con altri HTTP verb (es: POST)?
 - > Che status code puo' restituire? Cosa significano?

-> **Email:**

- > Quali sono gli elementi fondamentali del processo di invio/ricezione di email?
- > SMTP:
 - > Che cos'è e a cosa serve?
 - > Che well-known-port usa?
 - > È un protocollo testuale?
- > Flusso di invio:
 - > Descrivi gli step dell'invio di una email.
 - > In quali punti SMTP viene usato?
- > Struttura messaggi SMTP:
 - > Descrivi la struttura dei messaggi e le keyword usati.
 - > Hanno un sistema di status code?
- > SMTP vs HTTP:
 - > Usano connessioni persistenti?
 - > Che tipologia di protocollo sono?

-> **HTTP/2:**

- > Per quale motivo è stato inventato?
- > Quali sono le sue principali funzionalita' rispetto ad HTTP/1.1?
- > Usa TCP o UDP?

-> **HTTP/3:** Su cosa si basa? Usa TCP o UDP?

-> **Distribuzione file: P2P vs Client/Server:**

- > Che cos'è P2P?
- > Descrivi formalmente le tempistiche di distribuzione in Client/Server.
- > Descrivi formalmente le tempistiche di distribuzione in P2P.
- > Quale delle due è più scalabile?
- > Quale delle due richiede risorse sempre online?

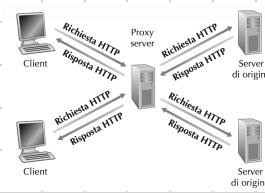
Proxy Web & Cache

Il proxy web è un intermediario che instrada le richieste web in reti private. Solitamente è fornito di cache.

• FUNZIONAMENTO

Alla ricezione di richieste dagli Hosts:

- Se cache hit → Restituisce l'elemento
- Se cache-miss → Chiama l'esterno e salva prima di restituire



In scenari di problemi di performance, soltamente è più conveniente usare il Proxy Web con cache.

• CONDITIONAL GET

Il proxy server richiederà periodicamente al web server se le risorse cache-ate sono scadute.

Questo tramite la richiesta al web server della stessa risorsa ma con l'header **IF-modified-since**

Il web server restituirà

- 304 (Not Modified): Se non ci sono mod.
- La risorsa richiesta altimenti

```

GET /fruit/kiwi.gif HTTP/1.1
Host: www.exotiquecuisine.com
If-modified-since: Wed, 9 Sep 2015 09:23:24
  
```

```

HTTP/1.1 304 Not Modified
Date: Sat, 10 Oct 2015 15:39:29
Server: Apache/1.3.0 (Unix)
(corpo vuoto)
  
```

EMAIL > WKT: 25

Il flusso generale di invio/ricezione/gestione email viene gestito da 3 attori principali

- USER AGENT: Es: Outlook, Gmail, Apple Mail..

Interface di interazione fra gli utenti e il mail server.

- MAIL SERVER:

Infrastruttura che consente l'effettiva ricezione e invio di messaggi. È richiesto per ogni utente con le mailbox

- SMTP: Simple Mail Transfer Protocol, in **TCP**

È il protocollo di comunicazione usato nello scambio di mail.

È presente sui mail server come (anche in contemporaneo):

- SMTP Client: Per gestire la ricezione di messaggi. (mittente)
- SMTP Server: == l'invio di messaggi (Destinatario)

• Flusso di esec m= mittente , d= destinatario

1. M usa lo user agent per comporre la mail. La invia
2. Lo user agent invia la mail nella coda di messaggi del mail server uscendo SMTP
3. Il mail server con il SMTP Client vede il messaggio in coda e apre una connessione TCP sul mail server.
4. I due mail server fanno un handshake SMTP, se ok il mail server invia il messaggio.
 - 4.a Se il mail server non è raggiungibile viene fatta una retry policy (ogni 30 min). Se fallisce, il mail server notifica m con una auto-email.
5. L' SMTP server riceve il messaggio e lo mette in mailbox.
6. Il mail server riceve il messaggio nella mailbox e □ eventualmente lo vedrà dello user agent

• ESEMPIO DI INVIO S=Server, C=Client

L'invio avviene riga per riga, con status

codes ad ogni statement.

- HELO '...' 250 Hello..., pleased: SMTP Handshake
- MAIL FROM: Specifica il sender. - RECIPIENT TO: Destinatario
- DATA: Specifica il body della mail. Finisce col ':'

È possibile dialogare direttamente con il browser tramite telnet. telnet serverName 25

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr ... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Ti piace il ketchup?
C: Che cosa ne pensi dei cetrioli?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

● HTTP VS SMTP:

	SMTP	HTTP
- CONNESSIONI PERSISTENTI	✓	SOLO da HTTP/1.1
- TIPOLOGIA DI PROTOCOLLO	PUSH	PULL

HTTP > HTTP/2

HTTP/2 è nato per ridurre la latenza percepita dell'utente

con:

● INTERLEAVING:

Suddivisione degli pacchetti HTTP in frame di uguale dimensione e trasmissione interallata per consentire il caricamento più veloce dei pacchetti più piccoli, insieme a compressione

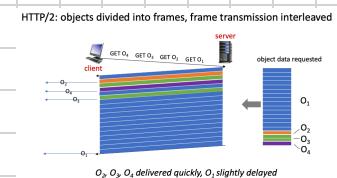
● PRIORITY:

Capacità di impostare priorità e dipendenze con pacchetti precedenti. (HTTP/1 era una coda normale FCFS)

● SERVERT PUSH: Invio di più risposte HTTP a fronte di 1 richiesta HTTP con info aggiuntive

HTTP > HTTP/3

Versione in sviluppo basata su UDP.



DISTRIBUZIONE FILE > P2P VS CLIENT SERVER

Il P2P è un'architettura di rete che non richiede la presenza di un'infrastruttura di server sempre attivi.

• ESEMPIO DI TRASFERIMENTO

Considerando la trasmissione di un file nelle 2 architetture con:

- F : la dimensione del file in bit.
- N : Il # di peer (Hosts) che richiede il file.
- u_s : La banda di upload (u_s per il server e u_i per i peer).
- d_i : // // download dei singoli peer

• CLIENT SERVER:

Il server deve trasmettere il file intero ad ogni peer, con un tempo di almeno: $\frac{N \cdot F}{u_s} \leftarrow \frac{\text{File} \times \# \text{Peers}}{\text{Banda di trasm}} \frac{(b)}{(b/s)}$

Allo stesso tempo, si considera il client con la banda minima $d_{min} = \min d_i$ per i peer

il quale impiega almeno: $\frac{F}{d_{min}} \frac{\text{Dim File}}{\text{Banda download}} \frac{(b)}{(b/s)}$

Quindi, con un tempo $t_s \geq \max \left\{ \frac{NF}{u_s}, \frac{F}{d_{min}} \right\}$
I. Distribuz: Server Client più lento
che cresce linearmente in funzione di N peers

• P2P

Il server trasmette la 1 volta il file alla comunità di peer con tempo di almeno: $\frac{F}{u_s}$

Analogamente a client-server, il client più lento richiede $\frac{almeno}{d_{min}} \frac{F}{u_s}$

Infine, in generale per trasmettere i file a N peers si considera la banda di upload totale (server+peers) con tempo di almeno $\frac{NF}{u_s + \sum u_i}$

con tempo $t_{p2p} \geq \max \left\{ \frac{F}{u_s}, \frac{F}{d_{min}}, \frac{N \cdot F}{u_s + \sum u_i}, u_i \right\}$
che cresce in maniera minore.

L07 >

-> **Email:**

-> Quali protocolli vengono coinvolti nella ricezione di email? Quali sono le loro caratteristiche?

-> **DNS:**

-> Come si puo' definire il DNS?

-> Qual'è il compito principale del DNS?

-> Qual'è la sua well-known-port?

-> Quale protocollo di trasporto usa sotto il cofano?

-> Quali altri compiti puo' gestire il DNS?

-> Descrivi tutti gli elementi della gerarchia dei DNS.

-> Che modalita' di risoluzione possono essere impiegate?

-> Caching:

-> Viene implementato il caching?

-> Un record cancellato/modificato viene propagato subito?

-> Cosa sono i resource-records?

-> Che tipi sono disponibili?

-> A cosa serve ogni tipo?

-> Descrivi tutti gli elementi della struttura di un resource-record.

-> Si possono mandare piu' interrogazioni in un messaggio DNS?

● PROTOCOLLI DI RICEZIONE EMAIL 2.3.4

SMTP è un protocollo di push, ovvero è pensato per il solo invio. La ricezione avviene con:

- POP3 · Semplificato
- IMAP: Più complesso, con ruletta gestione remota + cartelle
- HTTP: Dai web browsers

DNS > 2.4 Domain Name System WKP. S3

Il DNS è inteso come

- Database distribuito implementato in gerarchia di DNS Servers decentrati.
- Protocollo a liv. applicazione che consente di interrogare il Database.

Il suo compito principale è la traduzione di hostname in indirizzi IP, preferibilmente in UDP (o TCP per proxy).

● ALTRI COMPITI:

- HOST ALIASING: relay.westcoast.facebook.com → facebook.com

Traduzione di hostname sinonimi in canonici o IP

- MAIL ALIASING: bob@west.yahoo.com → bob@yahoo.com

Traduzione di mail domain sinonimi in canonici

- LOAD DISTRIBUTION:

Prevede la rotazione di più indirizzi IP che corrispondono allo stesso hostname. Utile su grandi siti (es: google).

● GERARCHIA DEL DNS

- Root SERVER: Forniscono gli IP dei TLD servers

- TLD SERVER: Top Level Domain

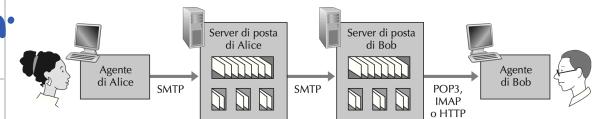


Figura 2.16 Entità comunicanti e protocolli per la posta elettronica.

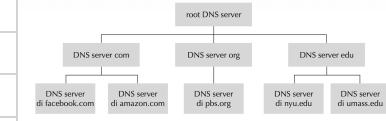


Figura 2.17 Gerarchia parziale di server DNS.

Si occupano dei domini di 1° livello (.com, .uk...) e

Forniscono gli IP degli Authoritative servers.

- AUTHORITATIVE

Ospita i record di associazione fra name e IP.

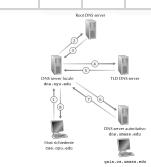
- LOCAL DNS: O Domain Name System

Server locali che forniscono IP agli Hosts e fanno da web proxy.

• TIPOLOGIE DI RISOLUZIONE

I ITERATIVA

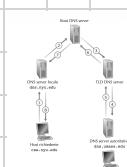
Il local DNS server sarà incaricato di comunicare con tutte le parti



I RICORSIVA

Il local DNS server si affida al Root DNS per la risoluzione.

- Overhead Root



• CACHING

Viene applicata a livello locale per salvare le traduzioni fissa

hostname-ip e bypassare i root DNS server. Ma dura 5 V

• RECORD DNS 24.3

I DNS server contengono Resource Records nella forma:

(Name, Value, Type, TTL) TTL = Time To Live cache

e il significato varia in base al Type

- Type = A : Name è un hostname e Value è l'IP di server Auton't.
- Type = NS : Name è un dominio e Value è hostname ✓ canonico
- Type = CNAME: Name // hostname e Value è l'hostname ✓ canonico
- Type = MX : // domain e // è il mailserver canonico.

• STRUTTURA RECORD DNS

Figura 2.21 Formato dei messaggi DNS.	
Identificazione	Flag
Numeri di domande	Numeri di RR di risposta
Numeri di RR autoritativi	Numeri di RR aggiuntivi
Sezione delle domande (numero variabile di domande)	- 12 byte
Sezione delle risposte (numero variabile di record di risorsa)	- Campi di nome e tipo di una query
Sezione autoritativa (numero variabile di record di risorsa)	- RR, in risposta alla query
Sezione aggiuntiva (numero variabile di record di risorsa)	- Record per i server autoritativi
	- Informazione aggiuntiva "d'aiuto" che può essere usata

Sia query che risposte dispongono della stessa struttura:

soppiata

1. IDENTIFICATION: Identifica la richiesta-risposta.
2. FLAG: Req/Res, Ricorsione, È il server autoritativo per il nome rich?
3. # query
4. # RISPOSTE (R.R.)
5. # AUTHORITY (R.R.)
6. # ADDITIONAL INFO (R.R)
7. QUESTIONS: Collezione di domande con nome e tipo
→ A, MX IP email
8. ANSWERS: Risposte ↗
9. AUTHORITY: RR. dei server autoritativi delle answers.
10. ADDITIONAL INFO: Come RR.

L08 >

-> **CDN:**

- > Cosa sono?
- > Quali sono le loro funzioni principali?
- > Quali solo le loro politiche di dislocazione? Che vantaggi e svantaggi presentano?
- > Si appoggiano a qualche protocollo?

-> **Streaming video:**

- > Definisci formalmente.
- > Vengono riprodotti a velocità costante o dinamica?
- > Cos'è il bitrate? Cosa succede quando è alto?
- > È applicato un layer di compressione? A che fine puo' essere applicato?
- > Che relazione bisogna preservare fra throughput e bitrate.
- > Come funziona lo streaming HTTP? Quali sono i suoi vantaggi e svantaggi?

-> DASH:

- > Che cos'è? Perché è stato inventato?
- > Su quale protocollo si basa?
- > Qual'è la sua unita' di misura?
- > Come viene gestito un video in streaming su DASH?
- > Quali sono i vantaggi che porta rispetto allo streaming HTTP?
- > Cos'è un manifest file? Cosa contiene?

CDN > Content Distribution Network

Le CDN sono reti di server distribuiti che cacheano contenuti in modo da abbassare la latenza e fare load balancing.

Le CDN possono avere come politiche di dislocazione

- ENTER DEEP: + veloce + manutenzione

Le CDN si dislocano il più vicino possibile agli Hosts

- BIGS HOME: - veloce - manutenzione

Le CDN si dislocano in pochi punti centralizzati vicino alle ISP Tier 1.

e funzionano sfruttando il DNS.

STREAMING DI VIDEO >

I video sono un insieme di immagini / frame visualizzati tipicamente a velocità costante.

- CARATTERISTICHE & METRICHE

- BIT RATE: È la velocità con cui viene trasmesso il video.

Maggiore è, migliore è la compressione. Può essere costante o var.

- COMPRESSEIONE:

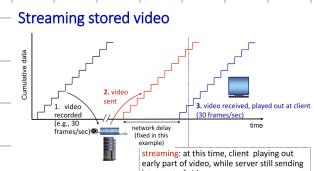
Viene applicata per avere un compromesso fra dimensione e bit rate. Consente di creare versioni con bit rate diversi.

L'importante è: throughput medio \geq bit rate.

- STREAMING HTTP: FUNZIONAMENTO

Il client stabilisce una connessione TCP e

richiede tramite HTTP un file video. Man mano che i byte del video vengono ricevuti, questi vengono memorizzati in un buffer a client-level. Superata una soglia, l'applicazione video



effettuerà il play.

Lo svantaggio è che non si possono cambiare le versioni del video durante lo streaming.

- STREAMING DASH: Dynamic Adaptive Streaming over HTTP.

È un protocollo di streaming basato su HTTP che risolve il problema di streaming su HTTP con:

- CHUNK:

Il video viene suddiviso in diversi chunk di alcuni secondi, ognuno in diverse versioni (bitrate).

Così il client potrà richiedere versioni diverse a seconda della banda disponibile in maniera dinamica.

- DISTRIBUTION

I chunk possono essere dislocati e replicati su diversi server nel mondo. Il singolo client richiederà un manifest file in HTTP per ottenere per ogni versione: bit rate e URL, e sarà in grado di cambiare versione e server nello streaming.

L10 >

-> **Transport Layer:**

- > Cosa mettono a disposizione dei processi?
- > Qual'è la differenza fra Transport layer e Network Layer?
- > Multiplexing/Demultiplexing
 - > Come funziona?
 - > Quali sono le criticità e in quale fase avvengono?
 - > Come viene determinato il socket con cui comunicare?
- > De-multiplexing:
 - > Come funziona in UDP? I pacchetti arrivano nello stesso socket?
 - > Come funziona in TCP? I pacchetti arrivano nello stesso socket?

-> **UDP:**

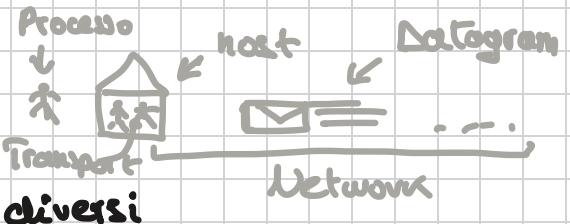
- > È un protocollo veloce? Come si comporta in condizioni di rete avverse?
- > Come funziona il flusso di invio/ricezione in UDP?
- > Struttura pacchetto UDP:
 - > Descrivi gli elementi della sua struttura.
 - > Ha una lunghezza fissa o dinamica?
- > Checksum:
 - > Che cos'è?
 - > Come viene calcolato?
 - > Da solo basta per poter fare error detection & correction?

TRANSPORT LAYER ➤ [Online learning da autore libro] L10

Il protocolli al livello 'Transport' mettono a disposizione una logical communication, ovvero un canale astratto (i 2 host potrebbero essere estremamente lontani fra loro).

• DIFF. FRA LIVELLI NETWORK E TRANSPORT

- Transport: comunicazione fra processi diversi
- Network: comm. fra host diversi. Il singolo datagram viene recapitato ad un host, che deciderà a quale processo venga inviato.



Quindi il lavoro del Transport è gestire i pacchetti inviati / ricevuti dal livello di Network con i processi. Network si occupa di invio/ricezione.

• MULTIPLEXING E DEMULTIPLEXING

Conceptualmente uguale agli altri livelli. Il livello Transport deve comprendere a quale applicazione e processo delle applicazioni in App. layer inviare il segmento, dato il datagram.

Questo si fa sfruttando gli header del livello Network, che contengono gli IP di source e dest. Combinati con le info.

sulle porte del transport, si potrà determinare il socket corretto.

DEMULTIPLEXING ➤ UDP & TCP

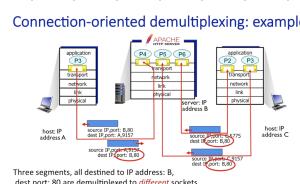
• UDP : Funzionamento semplice

Il sender crea un socket con una host-local port e invia il messaggio specificando una dest. port (2).

Il receiver (R) riceverà il messaggio e indipendentemente dall'IP del sender, il messaggio sarà recapitato sulla porta specificata.

In risposta, il receiver (R) risponderà invertendo host-local e destination ports.

• TCP



Siccome è connection oriented, TCP necessita distinzioni approfondite rispetto al solo num. di porta di UDP.

Ogni socket viene quindi identificato dal livello Transport come la quadrupla: (src. IP, src Port, dest IP, dest Port)

Questo consente a protocelli come HTTP di avere processi diversi che dialogano sulla stessa porta con proc. diversi.

UDP >

Anche se inaffidabile, la mancanza di connection, flow e congestion control lo rendono molto veloce. L'affidabilità può essere comunque implementata a livello applicativo (HTTP/3).

• FLUSSO UDP

Come prima, in fase di invio dal socket, verrà fatto il mux aggiungendo gli header. In fase di ricezione, viene fatto il demux e controllato il checksum header prima di fare forward al socket.

• SEGMENTO UDP

È formato da:

- Headers: 4 campi

a) Source Port b) Dest Port c) checksum

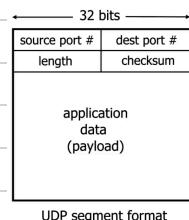
d) Length: In bytes su tutto il pacchetto, visto che il payload è variabile

- Payload

• CHECKSUM

Il checksum è un meccanismo di verifica degli errori.

Il suo funzionamento prevede di sommare tutti i dati relativi al pacchetto UDP (headers inclusi) e i source e destination IP



del Network datagram in 16 bit.

Le somme saranno fatte in complemento a 1 con somma di eventuale riporto.

- CORPL. 1 CON RIPORTO

Dati 2 num in BIN, si sommano normalmente e si tiene conto del riporto che poi si risomma e infine si invertono tutti i bit.

In fase di verifica, si riceva il checksum e si somma alle word iniziali, dovrebbe dare $\overbrace{111\dots1}^{16}$ se eretto.

NB: Il checksum è un metodo basico che non esclude totamente gli errori.

example: add two 16-bit integers
1 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0
1 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
wraparound
1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1
sum
1 0 1 1 1 0 1 1 1 0 1 1 1 1 0 0
checksum
0 1 0 0 0 1 0 0 0 1 0 0 0 0 1 1

Note: when adding numbers, a carryout from the most significant bit needs to be added to the result

L11 >

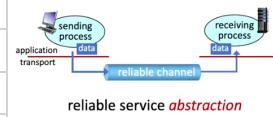
-> **Principi di RDT:**

- > Che tipologia di canale si vuole implementare fra sender e receiver concettualmente?
- > Come viene effettivamente implementato il canale fra sender e receiver?
- > Sono ammesse risposte dal receiver?
- > Quali sono le caratteristiche che deve prevedere RDT?
- > Gli elementi comunicanti hanno coscienza del loro stato?
- > Che API vengono usate da RDT?

-> **RDT:**

- > Descrivi come funziona RDT 1.0.
- > Descrivi come funziona RDT 2.0.
- > Descrivi come funziona RDT 2.1.
- > Descrivi come funziona RDT 2.2.
- > Descrivi come funziona RDT 3.0.

PRINCIPI DI RELIABLE DATA TRANSFER



ASTRAZIONE

Concettualmente, si vuole implementare un canale di comunicazione **unidirezionale** fra i processi sender / receiver

IMPLEMENTAZIONE

Avverrà attraverso un canale unidirezionale che ammette risposte dal receiver. Il protocollo deve prevedere

I Perdite I Bit Flippati I Dati con ordine diverso

Inoltre, va tenuto conto che gli attori non hanno coscienza dello stato dell'altro se non con la comunicazione di messaggi.

API DEL TRANSPORT LAYER

- rdt_send(): Invio dati da App layer.
- rdt_send(): Invio dati da Transport Layer nel Net. Layer.
- rdt_receive: Ricezione dati da Net. Layer su Tr. layer
- deliver_data: Invio dati verso App Layer.

• RDT Protocol viene definito attraverso le macchine a stati finiti:

1 per il sender e 1 per il receiver.

RDT > IMPLEMENTAZIONE

NOTA: Studiare diagrammi machine

RDT 1.0 :

Dato underlying comm. channel reliable,

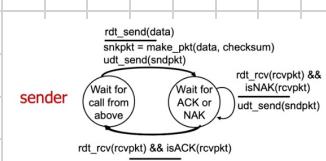
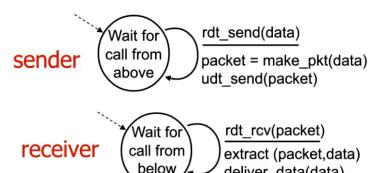
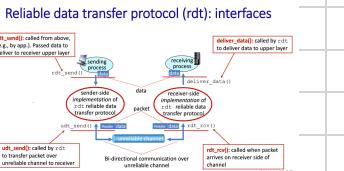
- FSM: SENDER : Farci mux + invio a receiver.
- FSM RECEIVER : Farci de mux + deliver ad App. Layer.

RDT 2.0 :

Dato underl. comm. channel **unreliable** (con bit errors nella trasm)

si usa un'architettura di Stop & Wait:

- FSM SENDER : Crea un checksum e aspetta

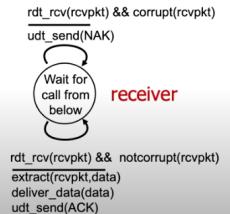


un ACK (continua) o NACK (re-invia) da receiver.

- FSM RECEIVER: Verifica il pacchetto:

→ **OK**: delivery ad App. Layer + risposta ACK

→ **CORR**: risposta NAK



RDT 2.1

Evoluzione che rileva ACK/NAK corretti tramite un seq. number

- FSM SENDER:

Creo un pacchetto con seq. num. Ø, per poi aspettare ACK/NAK:

→ ACK → Ripeto con seq. num 1, poi Ø e così via.

→ NAK o corrotto ⇒ Re-invio pacchetto con num. corrente

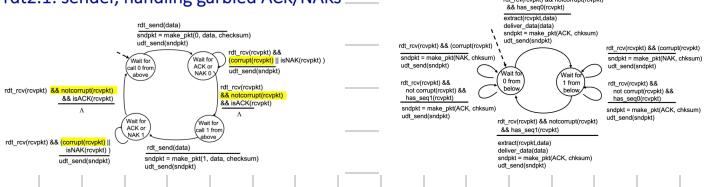
- FSM RECEIVER:

→ ≡ pkg corrotto ⇒ Invio NAK.

→ Arriva pkg Ø per wait 1 o vice-versa ⇒ Invio ACK

→ ≡ pkg Ø per wait Ø ⇒ Deliver + ACK

rdt2.1: sender, handling garbled ACK/NAKs



RDT 2.2:

RDT sviluppato in maniera da usare solo ACK mandando solo il pacchetto precedente

- SENDER FSM: Re-invia il pacchetto se

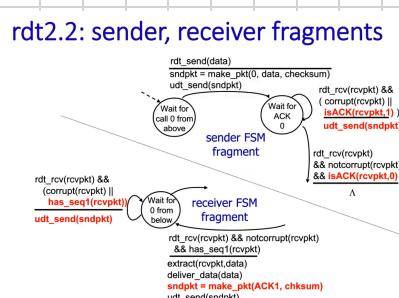
gli arriva un ACK con seq. diverso

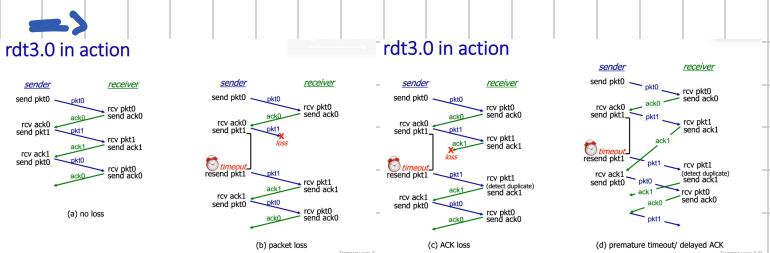
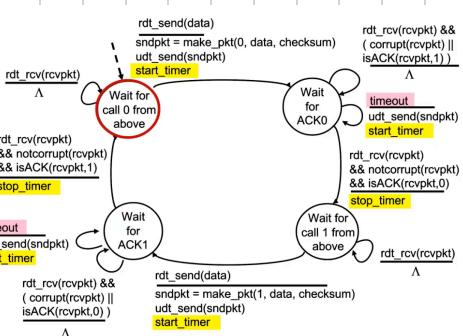
- RECEIVER FSM: Se il pacchetto è corrotto

o si riferisce ad un altro seq. num. ⇒ invia seq num. prec.

RDT 3.0: Copertura di pacchetti persi #END:Studio slide

In caso di pacchetti persi, viene messo un timer a livello sender che, quando scade, re-invia i pacchetti.





Casi possibili:

- Arrivo corretto
- PKT si perde => Timeout + Re-invio msg
- ACK si perde =>
- Timeout prem. / ACK in delay => Re-invio msg (scartato subito) + Invio msg succ

L12 >

-> RDT 3.0:

-> Ha dei rischi legati ai sequence numbers? Se si, come possono essere risolti?

-> Velocità:

-> L'RDT è veloce o lento?

-> Che cos'è e a cosa serve il Sender utilization? Come si calcola?

-> Pipelining:

-> Che cos'è?

-> Che problema risolve?

-> Di cosa necessita?

-> Go-Back-N:

-> Che cos'è e quali funzionalità offre?

-> Quali sono e cosa fanno i flag di cui si serve il sender?

-> Quali sono e cosa fanno i flag di cui si serve il receiver?

-> Qual'è la politica di gestione di timeout e ACK?

-> Che vantaggi e svantaggi porta?

-> Cosa succede se un receiver riceve un pacchetto out of order?

-> Selective Repeat:

-> Che cos'è e quali funzionalità offre?

-> Quali sono e cosa fanno i flag di cui si serve il sender?

-> Quali sono e cosa fanno i flag di cui si serve il receiver?

-> Che vantaggi e svantaggi porta?

-> Qual'è la relazione fra seq. numbers e la finestra di invio?

-> Cosa succede se un receiver riceve un pacchetto out of order?

-> TCP:

-> Che cos'è?

-> Usa comunicazione suplex, half-duplex o full-duplex?

-> Consente le comunicazioni multicast?

-> Fa uso di cumulative ACK?

-> Descrivi quali sono e cosa fanno i componenti di un segmento TCP.

ROT > ROT 3.1

L'ROT 3.1 ha ancora dei problemi

- DE-SYNC MSG:

Funziona fin quando viene mantenuta la sync fra sender/receiver con il seq. number (1 bit suscettibile a cond. rete). Si risolve usando seq. number con grande dim. (32 bit) in modo da minimizzare le prob. di desync.

- VELOCITÀ

L'ROT 3.0 è troppo lento per via dell'architettura stop & wait. Considerando infatti U_{sender} : La frazione di tempo impiegata a inviare dati come:

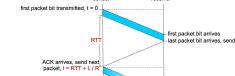
$$U_{\text{sender}} = \frac{L/R}{RTT + L/R}$$

- ESEMPIO:

Performance of rdt3.0 (stop-and-wait)

- U_{sender} : utilization - fraction of time sender busy sending
- example: 1 Gbps link, 15 ms prop. delay, 8000 bit packet
- time to transmit packet into channel: $D_{\text{trans}} = \frac{L}{R} = \frac{8000 \text{ bits}}{10^9 \text{ bits/sec}} = 8 \text{ microseconds}$

rdt3.0: stop-and-wait operation



$$RTT = 2 \cdot 1.8 \text{ ms}$$

rdt3.0: stop-and-wait operation

$$U_{\text{sender}} = \frac{L/R}{RTT + L/R} = \frac{8000 \text{ bits}}{30.008 \text{ ms}} = 0.00027$$

rdt 3.0 protocol performance stinks!

Protocol limits performance of underlying infrastructure (channel)

Si nota che sia molto bassa. È quindi necessario il pipelining.

- PIPELINING:

Prevede di inviare più pacchetti senza aspettare l'ACK. Richiede: più seq. numbers, buffer in sender e receiver.

Così l'utilization incrementerà linearmente, anche se rimarrà bassa.

PIPELINING > GO BACK N - Re-invio tanti prg + no buffer

È un protocollo di pipelining senza buffer.

- SENDER

Gestisce una finestra di N pacchetti "in-flight" tramite

→ send_base: Punti al pacchetto più vecchio non ancora ACK'd.

→ next seqnum: Punti al prossimo pacchetto inviabile nella finestra

- CUMULATIVE ACK: ACK di un pacchetto conferma i precedenti

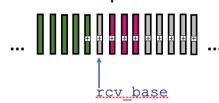


- TIMER: sul pacchetto più vecchio. Al timeout si ritrasmettono i pacchetti non ACK'd.

• RECEIVER

Usa il puntatore `rcv_base` per tenere traccia del prossimo pacchetto in ordine che si aspetta. In caso riceva pacchetti out-of-order, solitamente li scarta fino al timeout.

Receiver view of sequence number space:

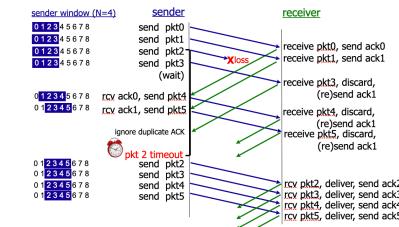


- received and ACK'd
- Out-of-order: received but not ACK'd
- Not received

• ESEMPIO : Sender window n

- Appena `pkz` viene perso, il receiver scarta tutti i pacchetti fino al timeout.

Go-Back-N in action



PIPELINING ➤ SELECTIVE REPEAT + Meno chiamate - Mem. su s/r.

Protocollo di pipelining che sfrutta buffer su sender / receiver per tenere traccia dei pacchetti e rimandarli il prima possibile.

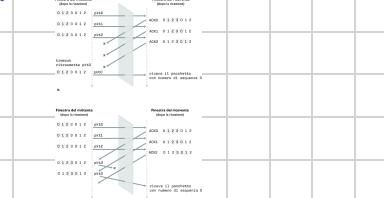
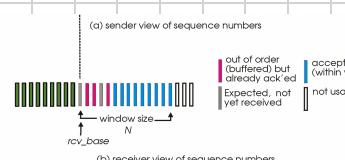
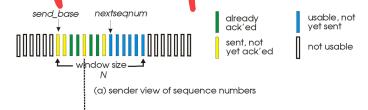
• SENDER!

Usa come prima `send_base` e `next seq num` per gestire una finestra di N pacchetti, in cui per ogni pacchetto terrà traccia sul suo ACK status e timer. Al timeout n-esimo, verrà ri-trasmesso solo quel pacchetto.

• RECEIVER

Faci ACK dei pkg arrivati correttamente, anche se disordinati. All'arrivo del pkg ordinato, manda ACK e fa delivery in ordine.

NB: $N \leq \frac{\text{seqnum}}{2}$ dove seqnum è lo spazio di seq num. Per funzionare



TCP >

È il protocollo di trasporto connection-oriented, full-duplex, point-to-point (no multicast) dei dati in maniera ordinata e affidabile. che fa uso di cumulative ACK con timeout sul pacchetto più vecchio.

non ACK'd e ritrasmissione del pacchetto che ha causato **timeout**.

• STRUTTURA:

1. SOURCE PORT

2. DEST. PORT

3. SEQ. NUMBER: Contenente il numero del 1 byte del pacchetto

4. ACK. NUMBER: \equiv il prossimo \downarrow che il receiver si aspetta.

5. LENGTH: Dell'intestazione TCP.

6. FLAG

7. RCV WND: Usato nel flow control per il # Bytes accettabili.
dal receiver.

8. CHECKSUM:

9. OPTIONS: Di lunghezza variabile

10. APP. DATA:

L13 >

-> **TCP:**

-> Timeout: Esplicita formalmente la formula usata per calcolare il timeout

-> Che cos'è e a cosa serve Fast Retransmit?

-> Flow Control:

-> Che cos'è?

-> Da che campo TCP viene gestito?

-> Che campo gestisce la dimensione generale del buffer? È modificabile?

-> Handshake:

-> Che cos'è e come funziona?

-> Definisci i passaggi di handshake con i flag utilizzati e i loro valori.

-> Chiusura connessione:

-> Come viene gestita la fine di una connessione TCP? Spiega i passaggi e flag usati.

-> Congestion Control:

-> Quali sono i fattori che contribuiscono alla congestione?

ESEMPIO D'USO ACK / SECE. NUTRI:

- Il receiver fa echo del msg
- Il sender popola ACK per sync. con receiver.

STURA DEL TIMEOUT

Il timeout deve essere più grande dell'RTT ma non troppo.

Per il calcolo si usa la formula:

$$T.I = E.RTT + \alpha \cdot \text{DEV.RTT} \quad \text{Con } \alpha \text{ (default=4), } \beta \text{ v. costanti}$$

- S.RTT : È la misura dell'RTT, senza cont. retrasm.
- E.RTT è una media mobile che applica un peso storico:

$$E.RTT = (1-\alpha) \cdot E.RTT + \alpha \cdot S.RTT$$

- DEV.RTT : È la varianza dell'RTT, usata come 'margine'

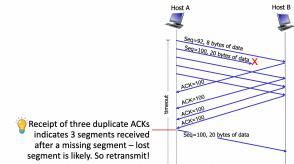
$$\text{DEV.RTT} = (1-\beta) \cdot \text{DEV.RTT} + \beta |S.RTT - E.RTT|$$

TCP fast retransmit

FAST RETRANSMIT

Funzionalità TCP che forza la ri-trasmissione

di un pacchetto a cui sono arrivati 3ACK duplicati.



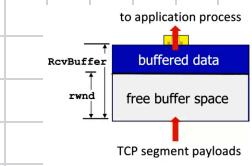
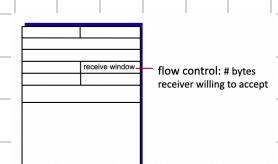
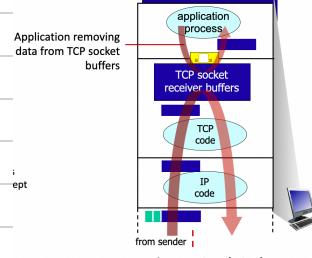
FLOW CONTROL >

Lato receiver, concettualmente il livello Network consegnerà i segmenti nel TCP Buffer, lo stesso dal quale poi prenderanno i protocolli applicativi.

Il flow control consente di impostare limiti per evitare overflow del Buffer e sono regolati dal campo receive window scambiato nei pacchetti.

RECEIVE WINDOW: RWD

Calcolato come spazio ancora disp. nel buffer.



Aggiornate manualmente. La dim. generale del buffer (Raw Buffer) è aggiustabile tramite i settaggi socket (4096b)

HANDSHAKE >

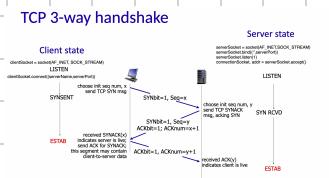
Viene effettuato prima della trasmissione in three way per richiedere e configurare la connessione.

- 1) Sender richiede conn.
- 2) Receiver manda conferma conn
- 3) = manda conf. ricezione della conf. del receiver

FLUSSO/IMP

Si usano i flag SYN e ACKBit

1. SENDER: $SYN=1$ e SEQ NUMBER = x generato casualmente.
2. RECEIVER: $SYN=1$, SEQ NUMBER = y \swarrow \searrow
 $ACKBit = 1$ e ACK NUMBER = $x+1$
3. SENDER: $ACKBit = 1$ e SEQ NUMBER = $y+1$



TCP > CHIUSURA

Avviene con uno scambio di segmenti con flag FIN:

1. SENDER: Invia un segmento con Flag FIN
2. RECEIVER: Risponde con ACK e invia un altro pacchetto di FIN
3. SENDER: Risponde con ACK

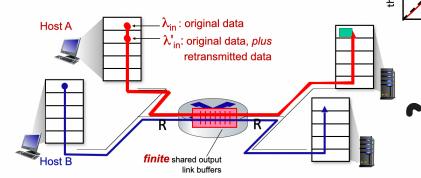
TCP > CONGESTION CONTROL

La congestione è causata sia da rallentamenti propri (retransmission + ACK...) che da terzi (altri elementi trasmettono sui router). Quando si approccia il PQA si vede una sola diminuzione della velocità.

Causes/costs of congestion: scenario 2

Realistic scenario: un-needed duplicates

- packets can be lost, dropped at router due to full buffers – requiring retransmissions
- but sender times out prematurely, sending two copies, both of which are delivered



L14 >

-> Intro su Congestion Control:

-> End-end:

- > Prevede una gestione interna al Transport Layer oppure esterna?
- > Come vengono rilevate le congestioni?

-> Network-assisted:

- > Prevede una gestione interna al Transport Layer oppure esterna?
- > Come vengono rilevate le congestioni?

-> Solitamente, quale viene preferita? Perchè?

-> Panoramica AIMD:

- > Da chi è usato?
- > Su cosa è basato?
- > Per cosa sta A.I. e come funziona?
- > Per cosa sta M.D. e come funziona?
 - > Quali sono le variazioni in TCP Tahoe?
 - > Quali sono le variazioni in TCP Reno?
 - > Qual'è lo svantaggio principale di AIMD?

-> **Funzionamento AIMD**:

- > Viene usato un campo per la gestione della AIMD? Se si quale? Ci sono dei vincoli?
- > Che cos'è e come viene calcolato il TCP Sending Rate?

-> **Algoritmo AIMD**:

- > Cosa succede a inizio comunicazione?
- > Che cos'è il TCP sending rate? Come viene calcolato?
- > Slow start:
 - > Che cosa succede?
 - > Come finisce?

-> Congestion Avoidance:

- > Cosa succede?
- > Come finisce?

-> Fast Recovery:

- > Cosa succede?
- > Come finisce?

-> **TCP Cubic**: Che cos'è? Come funziona?

-> **Delay based CC**:

- > Che cos'è?
- > Su quali componenti si basa?
- > Come funziona?

-> **Net. Assisted CC**:

- > Che cosa succede a livello Transport?
- > Che cosa succede a livello Network?
- > TCP è fair? Se si lo è sempre?

CONGESTION CONTROL >

La gestione del congestion control avviene tramite:

- END-END C.C.

Gestione interna al Transport Layer, in cui le congestioni vengono rilevate tramite misurazioni sul comportamento della rete.

- NETW.-ASSISTED C.C.

Gestione assistita dal Network Layer, il quale fornisce feedback tramite i router intermedi. I router possono informare se sono congestionati oppure la frequenza trasmissiva che possono supportare.

Soltanto si usa la 1° perché non tutti i router sono compatibili.

- AIMD:

È l'approccio di congestion control usato da TCP, basato dall'end-end congestion control. Regola il sending rate (Tasso d'invio) con:

- ADDITIVE INCREASE:

Incremento di $\frac{1}{RTT}$ ogni RTT fino ad una loss.

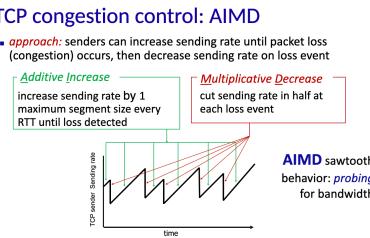
- MULTIPLICATIVE DECREASE:

Decremento della perdita dei pacchetti. Varia in base alla versione:

→ TCP Tahoe: (versione iniziale) scende a 1 MSS **sempre**.

→ TCP Reno: (\leq attuale) dimezza l'MSS dopo 3 ACK uguali e scende a 1 MSS quando ci sono timeout.

- Svantaggio: Su congestioni multiple dalla durata breve, si generano tempi sprecati di congestione.

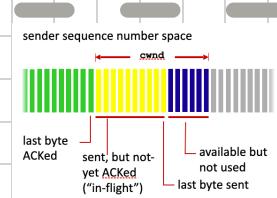


AIMD > FUNZIONAMENTO

Viene usato un campo cwnd (Congestion Window) \geq rwd

che verrà aggiornato dinamicamente in base alle osservazioni sulla rete

Impone il vincolo: Last Byte Ack - Last Byte Sent \leq cwnd



in cui il sender dovrà aspettare se superato. Questo vincolo regola quindi il sender rate come: $\text{TCP Sending Rate} = \frac{\text{cwnd}}{\text{RTT}} \text{ b/s}$

AIMD > ALGORITMO PROPRIO

1. SLOW START:

$$\text{Sending Rate} = \frac{\text{MSS}}{\text{RTT}}$$

La comunicazione inizia con $\text{cwnd} = 1 \text{ MSS}$. Ad ogni ACK ricevuto viene incrementata di 1 MSS . Può terminare:

A. TIMEOUT: Si impostano le variabili cwnd e slow start threshold:

$$\text{ssthresh} = \text{cwnd}/2 \quad \text{cwnd} = 1 \text{ MSS}$$

e si ricomincia slow start

B. $\text{cwnd} \geq \text{ssthresh}$: Passaggio a congestion avoidance:

C. 3 ACK DUPLICATI: $\text{ssthresh} = \text{cwnd}/2$, $\text{cwnd} = \text{ssthresh} + 3 \cdot \text{MSS}$ → fast recovery

2. CONGESTION AVOIDANCE:

$\text{cwnd} = \text{cwnd} + \text{MSS}$ (incremento di 1 MSS), ogni RTT

Termina per:

A. TIMEOUT: Uguale a slow start.

B. 3 ACK DUPLICATI: uguale a slow start

TCP CUBIC

- K : point in time when TCP window size will reach W_{\max}
 - x : start of timeout
 - Increase W as a function of the **cube** of the distance between current and K
 - Large increases when further away from K
 - Smaller increases (cautious) when nearer K
 - TCP CUBIC default in Linux, most popular TCP for popular Web servers
-

3. FAST RECOVERY

$\text{cwnd} = \text{cwnd} + 1 \text{ MSS}$ ad ogni ACK fino a che i pacchetti persi non vengono recuperati. Termina per

A. ACK PER PACCHETTO PERSO: Passaggio a congestion avoidance.

B. TIMEOUT: uguale a slow start.

TCP CUBIC >

Variante di AIMD, che cambia la gestione del cong. control.

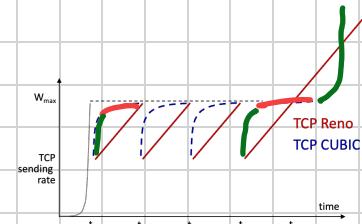
Dato

- W_{\max} : cwnd dell'ultima congestione:

- t: Il tempo corrente
- K: L'istante futuro in cui $cwnd = W_{max}$

Si prevede di:

1. aumentare di molto cwnd fin quando è lontana da W_{max}
2. // di poco cwnd // è vicina da W_{max}



DELAY BASED C.C. >

Metodo alternativo che si basa su RTT e Throughput misurato.

- Throughput misurato = $\frac{\# \text{ Byte inviati}}{RTT_{\text{misurato}}}$
- RTT_{\min} : RTT minore misurato
- Throughput non congestionato: $\frac{cwnd}{RTT_{\min}}$

A seconda del T. misurato cambia la gestione di cwnd:

- T. misurato è vicino al T. non congestionato:

Si aumenta linearmente cwnd

- T. misurato è lontano al T. non congestionato:

Si decrementa linearmente cwnd.

NETWORK ASSISTED C.C. >

Si basa sullo scambio di informazioni a livello:

- Transport: Tramite flag ECE nell'ACK del receiver.
- Network: Tramite flag ECN nei datagrammi IP impostati dai router durante l'invio.

FAIRNESS >

TCP è fair solo se:

- RTT è lo stesso
- Il numero di sessioni TCP è lo stesso
- Solo in congestion avoidance.

L16 >

-> **Divisione del network layer:**

- > Come viene diviso il network layer?
- > Come puo' essere descritto il funzionamento del network layer?

-> **Struttura del router:**

- > Descrivi tutti i componenti del router e cosa fanno.

-> **Input port:**

- > Che funzionalita prevede?
- > Quali sono i due modi con cui si puo' fare "switching" di un pacchetto?

-> **Longest Prefix Matching:**

- > Che cos'è e come funziona?

-> **Switching Fabric:**

- > Che cos'è?
- > Come viene implementata?

NETWORK LEVEL >

Il livello network è diviso fra:

- DATA PLANE: # forwarding

Parte che determina come un pacchetto in arrivo nella input port del router viene "forwardato" nella output port corretta.

Il forward avviene attraverso verifiche tra il datagramma e la routing table del singolo router.

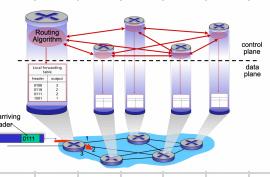
- CONTROL PLANE: # Routing

Parte logica generale che determina il percorso seguito dai datagrammi nella comunicazione.

Sfrutta un algoritmo di routing, che andrà poi a popolare le routing tables dei singoli router nel data plane.

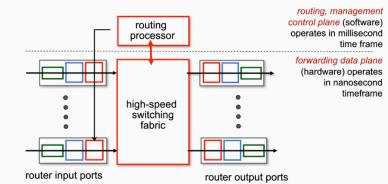
NB: Il funzionamento del livello network è best-effort

ed è una cosa voluta per essere semplice



Network Architecture	Service Model	Quality of Service (QoS) Guarantees ?			
		Bandwidth	Loss	Order	Timing
Internet	best effort	no	no	no	no
ATM	Ambient Bit Rate	guaranteed	yes	yes	yes
Internet	Voice/Guaranteed	yes	yes	yes	yes
Internet	Video (VC-103)	possible	possibly	possibly	no

high-level view of generic router architecture:



ARCHITETTURA ROUTER >

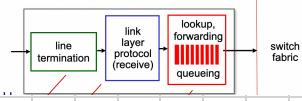
Concettualmente, il router sposta i pacchetti in arrivo da una porta di input a quella out.

- IN PORTS: Dove arrivano i pacchetti
- SWITCHING FABRIC: Componente HW che collega le in/out porte
Ad alta velocità per evitare congestioni.
- OUT PORTS: Dove escono i pacchetti.
- ROUTING. PROCESSOR: Esegue le logiche di Routing (Control Plane)

INPUT PORT >

L'input port si occupa di gestire le funzionalità di:

- LAYER PHYSICAL: Ricezione dei bit
- LAYER L2/L3: Gestione ed elaborazione frame



- DECENTRALIZED SWITCHING:

memoria di input port

Determina la porta di dest. basandosi sulla Fwd Table el. 1.

- DESTINATION BASED FWD: Basato su IP del destinatario

- GENERALIZED FWD: Basato sull' intero contenuto del datagram.

DESTINATION BASED FORWARDING >

Siccome gli IP hanno 32 bit (v4) o 64 bit (v6), non si può

definire una tabella che contenga ogni IP.

Bisogna definire dei range che siano anche

flexibili. Attualmente si usa il longest Prefix

Matching, che prevede di fare forward solo sul match più lungo.

ES:

Destination Address Range	Link interface
11001000 00010111 00010000 00000000	0
11001000 00010111 00010111 11111111	3
11001000 00010111 00010000 00000100	
11001000 00010111 00010000 00000111	
11001000 00010111 00010100 00000000	1
11001000 00010111 00010111 11111111	2
11001000 00010111 00011001 00000000	
otherwise	3

examples:
 1 11001000 00010111 00010110 10100001 which interface?
 2 11001000 00010111 00011000 10101010 which interface?

$$\begin{matrix} 1 \Rightarrow \emptyset \\ 2 \Rightarrow 1 \end{matrix}$$

Forwarding table	
Destination Address Range	Link Interface
through	0
11001000 00010111 00010111 11111111	
11001000 00010111 00010000 00000100	3
11001000 00010111 00010000 00000111	
11001000 00010111 00011000 00000000	1
11001000 00010111 00011001 00000000	2
otherwise	3

La ricerca del longest prefix deve essere pesantemente ottimizzata

sia come HW che con cache e algoritmi per stare nei nanosec.

SWITCHING FABRIC > IMPLEMENTAZIONE

La switch fabric viene implementata come:

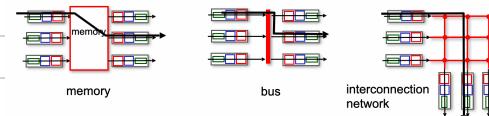
- MEMORY CORE: I/O con memoria intermedia. Lento.
- BUS CORE: Usa un collegamento diretto con 1 Bus. Ha cattiva.
- INTERCONNECTED CORE: Usa una matrice di bus. Usata attualmente.
- SWITCH RATE

Tasso di trasferimento da input a out

port. Idealmente deve essere $N \cdot R$ dove:

- N: #Inputs

- R: Capacità fisica di ricezione.



L17 >

-> **Buffering:**

-> Quali sono gli eventi che influenzano il buffering?

-> Che cos'è l'HOL delay?

-> Buffer Size:

-> Come viene determinata la buffer size?

-> Cosa succede per buffer size grandi?

-> Descrivi tutte le politiche di gestione dei buffer.

-> Net Neutrality:

-> Che cos'è?

-> Cosa protegge?

-> Quali sono i principi fondamentali?

-> **IP:**

-> Descrivi tutti i campi del protocollo IP.

-> Frammentazione:

-> Che cos'è?

-> Come funziona?

-> Che campi IP usa per funzionare?

-> Come vengono calcolate le variabili di frammentazione?

-> IPV4 Addressing:

-> Com'è formata la struttura IPV4 dei campi?

-> Che cos'è una Network Interface? Come si integra con l'addressing?

-> Che cos'è una subnet? da cosa è formata?

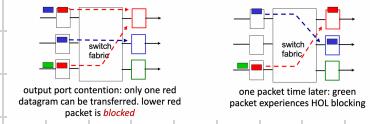
-> Descrivi le classi di subnetting

-> Quali sono gli indirizzi speciali nel subnetting?

-> Che differenza c'è fra subnetting e supernetting?

Eventi che richiederanno il buffering dei pacchetti sono:

- INPUT PORT QUEUING: Input port più veloce di switch fabric.
- HOL DELAY: Head Of Line Delay



Tipo di delay causato dall'attesa del pacchetto in cima alla coda

- OUTPUT PORT QUEUING: Switch fabric invia più velocemente di output port

BUFFER SIZE

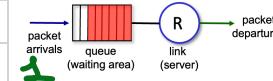
Dato C : capacità del link, N : # di flussi TCP indipendenti:

- $BS = C \cdot RTT$: Inizialmente
- $BS = \frac{RTT \cdot C}{\sqrt{N}}$: Dopo nuovi studi

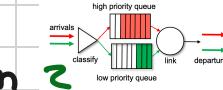
Bisogna tenere i buffer non troppo grandi per non incrementare V.

POLITICHE DI GESTIONE DEI BUFFER:

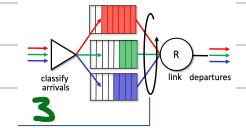
1. FCFS: first Come first Served.



2. PRIORITY: Definizione di priorità dai campi el Datagram



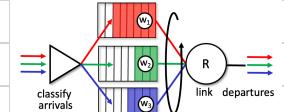
3. ROUND ROBIN: Vengono determinate delle classi e i pacchetti vengono accodati alternatamente



4. WEIGHTED RR: Variante di R.R. in cui ogni classe

ha un peso w_i ed avrà una "quota"

pari a: $\frac{w_i}{\sum w_j}$



NETWORK NEUTRALITY

Eventuali drop policy o code di priorità devono tenere conto dei principi di net neutrality, ovvero le leggi che tutelano principi socio-economici (es: Free speech, Enforce legal rules). Le legislazioni USA/EU definiscono che le reti devono essere pubbliche.

- NON-BLOCKING: Non bloccare siti legittimi
- NON-THEFTING: / rallentare / penalizzare il "
- NO Paid Priority:

PROTOCOLLO IP >

• STRUTTURA

1 IP VERSION: V4 o V6

2 HEADER LENGTH

3 TYPE OF SERVICE: Servono a distinguere i diversi tipi di datagrammi

4 LENGTH: Definisce la lunghezza dell'intero datagramma.

max: 65KB, media: 1500B per non effettuare
frammentazione (IPV6 non lo fa).

5. 16BIT ID, FLAGS, FRAGMENT OFFSET: Usati nella fragementazione.

6. TTL: Campo usato per evitare che i pacchetti continuino
a circolare. Ad ogni passaggio viene decrementato, se
raggiunge 0, il datagramma viene scartato.

7. PROTOCOL: Usato per determinare a quale transport layer
protocol consegnare il datagramma (simile port. number).

8. CHECKSUM:

9. SOURCE

10. DEST

11. OZIONI

12. DATI

FRAGMENTAZIONE V2 >

Nasce dal fatto che le MTU dei frame (~1500 Byte) sono
inferiori rispetto alla dimensione massima del pacchetto IP.

• FUNZIONAMENTO

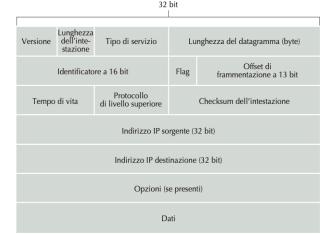


Figura 4.16 Formato dei datagrammi IPv4.

Il frammento viene scomposto in più frame a cui viene aggiunto in intestazione altri campi come offset e altri flag

I pacchetti vengono poi riassemblati solo alla destinazione.

In cui il destinatario avverrà un timer, se scade -> scarso tutto.

• CAMPPI IP DI FRAMM.

- LENGTH : Lunghezza di tutto il frame (20 Byte header IP)
- ID : Identifica i frammenti dello stesso pacchetto orig.
- FRAGFLAG : 1 se non è l'ultimo frammento, 0 altrimenti
- OFFSET : Posizione logica a blocchi di 8 byte.

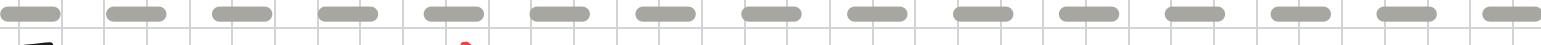
IP fragmentation/reassembly

• PROCEDURA DI FRAMM.

1. Dividere Len. Datagram/MTU per avere

frammenti.

2. Ogni frammento avrà offset $N = (\text{RTU} - 20\text{Byte}) / 8$ con

$$\text{Offset} = \text{offset} \cdot n$$


IPV4 ADDRESSING >

Gli indirizzi IPv4 sono formati con 32 bit in notazione decimale e sono associati ad una interfaccia di rete.

dotted-decimal IP address notation:
223.1.1.1 = 11011111 00000001 00000001 00000001
223 1 1 1

• NETW INTERFACE: Collegamento router/switch e il comm. link.

• SUBNET:

Rete di dispositivi che può collegarsi senza router. I router collegano subnet diverse e hanno 1 netw. Interface per ogni subnet.

• STRUTTURA SUBNET

Ogni subnet è divisa in network e host parts per distinguere

La notazione per distinguerle è:

xxxx.xxxx.xx,xx.xxxx/x / B

Dove:

- **B**: È il # bit dedicati alla rete
- **32-B** : Sono i bit dedicati agli hosts.
L'ultimo (255) è dedicato per le comunicazioni in broadcast.
Il 1° è usato per identificare la rete.

• CLASSFUL ADDRESSING: CLASSI

- A: Usano net. id da 8 bit [1, 126]
- B: // / / 16 bit [127, 191]
- C: // / / / 24 bit [192, 223]
- D: Riservate a multicast [224, 239]
- E: Riservate ad usi futuri [240, 255]

• INDIRIZZI SPECIALI

- | Net | host | |
|-------------------|--------------------------|---------------------------------------|
| - 0...0; 0..0 | : Questo host senza IP | - ∞ 1...1: Broadcast da router |
| - 0...0; ∞ | : Un host su questa rete | - 127...10..0: Localhost |
| - ∞ ; 0..0 | : Net id | - 1...11..1: Broadcast host |

• IP ADDRESSING

È possibile effettuare il

- **SUBNETTING**: Uso n bit in più dalla host part per dividere una subnet in 2^n
- **SUPERNETTING**: Uso N bit in più dalla network part per unire 2^N subnet in una supernet.

Questo funziona tramite le VLSM che consentono più subnet mask dentro la stessa rete.

L18 >

-> IP Addressing:

- > Che cos'è CIDR?
- > Quanti host si possono creare con una rete /N?
- > Che cos'è una subnet mask?
- > Come posso verificare se un IP appartiene a una data subnet?
- > Quali sono gli IP privati?

-> DHCP:

- > Che cos'è? A cosa si pone come alternativa?
- > Spiega come funziona il processo di DHCP, insieme alla struttura dei messaggi DHCP
- > Come vengono distribuiti gli IP agli ISP?

NETWORK > CIDR

Il **Classless InterDomain Routing** è la notazione usata in IPM
 $32 - \underline{B}$

- **# HOSTS**: $2^{32-B} - 2$: \pm Per identificare rete e \pm Broadcast.

- **SUBNET MASK**: Verifica IP in subnet: AND con SM

Modo alternativo per definire i bit assegnati alla subnet.

Assegna i bit della subnet a \pm e li rappresenta in binario.

ESEMPIO: $10.x.x.x/18$: $10.x.x.x/255.0.0.0$

- **IP PUBBLICI / PRIVATI**

Gli IP pubblici possono dialogare con l'esterno, mentre gli IP privati necessitano altri strumenti. Sono subnet private:

| 10/8 | 172.16/12 | 192.168/16 | 127/8 → IP localhost

FORWARDING DEI PACCHETTI >

Ogni host al suo interno ha una routing table che consente di mappare gli host vicini della stessa subnet e il gateway a diverse interfacce di rete

DISTRIBUZIONE DEGLI IP >

Gli IP possono essere impostati negli Hosts:

- **MANUALMENTE**: Possono generare errori (es: Stesso IP router).
- **DINAMICAMENTE**:

→ IN UDP.

Tramite il protocollo DHCP (Dyn. Host Conf. Prot). Funzionalmente, il DHCP comunica informazioni fondamentali per la conn.

- **FUNZIONAMENTO DHCP**

1. Il nuovo host richiede in broadcast nella stessa sottorete.

se esiste un server DHCP. (DHCP Discover)

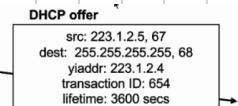
1.1 L'Host riempie il messaggio DHCP come

- source: 0.0.0.0:68 → IP non ancora assegnato

- dest: 255.255.255.255: 67: Broadcast
- yiaddr: 0.0.0.0 → Your Internet Address non ancora assegnato
- transaction ID: ∞ casuale → Usato per identificare l'host

2. Il DHCP server risponde in broadcast (DHCP Offer)

assegnando in yiaddr una proposta di IP addr.
con lo stesso transaction ID e una lifetime



3. L'Host risponde in broadcast con una nuova transaction

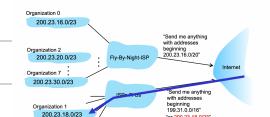
la richiesta d'uso dell'IP (DHCP Request)

4. Il DHCP server da un ACK in broadcast rispondendo alla transaction della DHCP Request

DHCP mette a disposizione meccanismi di rinnovo dell'IP.

DISTRIBUZIONE IP > PARTE DI RETE

La distribuzione degli IP avviene tramite la ICANN, che li
alloca in dei registri regionali. Questi saranno poi divisi
fra le ISP che a loro volta divideranno le loro reti
allocando altri network bit.



L19 >

-> **NAT**:

- > Che cos'è?
- > Come funziona?
- > Da chi viene usato?
- > Descrivi la struttura dei record NAT Tables.

-> **ICMP**:

- > Che cos'è?
- > A cosa serve?
- > Come si relaziona con IP?

-> **IPV6**:

- > Che cos'è?
- > Qual'è la sua struttura?
- > Si puo' comunicare in reti IPV4? Se sì, come funziona la comunicazione?

-> **Generalized Forwarding**:

- > Cos'è e cos'è il pattern match-action?
- > Descrivi cosa sono le Flow Table e il loro contenuto, includendo i campi
- > Le flow table riguardano piu' livelli della pila protocollare? Quali?
- > Quali sono le action possibili?

-> **Middlebox**:

- > Cosa sono?
- > Come sono classificabili?

NAT > NETWORK ADDRESS TRANSLATION

Il NAT è un servizio del router che abilita la comunicazione fra reti pubbliche (WAN) e private (LAN)



Funzionalmente avviene tramite il router stesso, che appare all'esterno come un singolo host rappresentante la rete privata.

• FUNZIONAMENTO

Si serve delle NAT Translation Tables, che mappano la coppia <IP router: porta> con <IP privato: porta> della rete.

- Ricezione pubblica → privata: le reti esterne vedranno l'IP del router. Il router sostituirà l'IP corretto e inoltrerà
- Invio privata → pubblica: Gli host mandano i pacchetti al router, che sostituirà l'IP di origine con il suo per ricevere

ICMP >

È un'estensione del protocollo IP usato per la gestione e notifica di errori e info. di gestione del protocollo IP stesso. È obbligatorio implementarlo insieme a IP e viene gestito da IP nel payload.

- CATEGORIE: 1. Errori 2. Richieste / Risposte info. ICMP del datagr. IP causa
- CAMPI: 1. Type : Tipo errore 2. Code: Dettaglio 3. Primi 8 byte

IPV6 >

Nuova versione creata perché gli indirizzi IPv4 sono esauriti.

• STRUTTURA:

1. VERSIONE

2. CLASSE DI TRAFFICO: simile a Type of Service.

3. ETICHETTA DI FLUSSO: Identifica un flusso di datagrammi

4. PAYLOAD LENGTH

5. INTESAZIONE SUCCESSIVA: Uguale a protocollo

Versione	Classe di traffico	Etichetta di flusso	32 bit		
Lunghezza del payload	Indirizzo sorgente (128 bit)	Intestazione successiva	Limite di hop		
			Indirizzo destinazione (128 bit)	Dati	

G. **HOP LIMIT**: uguale a TTL

Z. **IP SRC**: 128 bit

8. **IP DESC**: // //

9. **PAYLOAD**:

Di per sé sono molto più snelli, anche se in presenza di link piccoli ritorneranno errore.

• COMUNICAZIONE V4-V6

Le reti IPv6 sono retrocompatibili nelle comunicazioni IPv4 (anche se diverse), la stessa cosa non vale in IPv4. Avviene:

- Con Hosts che hanno sia IPv4 che IPv6, in cui l'eventuale scelta di quale IP usare viene dal DNS
- Attraverso il Tunnelling.

L'idea del Tunnelling prevede di usare il contenuto di un pacchetto come payload di un altro pacchetto.

Questo abilita l'IPv6 ad essere incapsulato in IPv4.

• TUNNELLING: COME FUNZIONA

All'arrivo di un datagramma da V6 a V4 al router di frontiera, si crea un nuovo pacchetto con source IP il router di frontiera e destination IP l'ultimo router contiguo con interfaccia V4-V6.

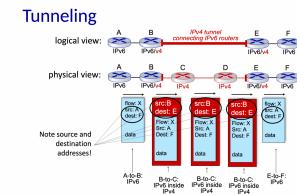
GENERALIZED FORWARDING >

Segue il pattern "match-action" tramite

- La definizione dei campi su più livelli per il match.
- L'action da effettuare come reazione.

Le quali sono definite tramite delle tabelle dette flow table, le quali vengono calcolate e distribuite da un controller remoto.

• FLOW TABLE



• CONTATORI: Insieme di contatori contenenti informazioni riguardo l'istanza della flow table.

• MATCH:

Describe:

→ PORTA D'INGRESSO

→ CAMPI DI LINK LAYER: SRC MAC, DEST MAC, ETH TYPE, VLAN ID, V.^{ULAN ID}

→ CAMPI DI NETW. LAYER: IP SRC, IP DST, VERSION, TOS

→ == TRANSPORT //: TCP/UDP SRC PORT, TCP/UDP DEST PORT.

• ACTION:

Suddivisibili come:

→ FORWARD: verso una porta specifica, in broadcast, in multicast o al controller remoto.

→ DROP

→ MODIFY: Modifica dei campi in match.

MIDDLE BOX >

Sono qualsiasi dispositivo HW o SW che svolge funzioni diverse da IP nei percorsi di rete fra origine e destinazione.

Sono categorizzabili come:

• NAT:

• SECURITY: come i firewall

• PERFORMANCE: come cache, load balancing, ecc.

L20 >

-> **Intro Control Plane:**

- > In che modi puo' essere implementato?
- > Come viene determinata e gestita la routing table?

-> **Intro ad algoritmi di Routing:**

- > Quali sono le classificazioni possibili per gli algoritmi di routing?

-> **Algoritmo di Djkstra:**

- > Com'è classificabile come algoritmo?
- > Come funziona?
- > Cosa sono i Link state broadcast e a cosa servono?
- > Qual'è la sua complessità temporale?
- > Cosa sono le oscillazioni? Come si possono evitare?

-> **Algoritmo Bellman-Ford:**

- > Com'è classificabile come algoritmo?
- > Come funziona?
- > Quando viene triggerato?
- > Su cosa si basa?

CONTROL PLANE >

Il routing del control plane può essere implementato come:

- Per-router Control:

Le operazioni di data/control plane vengono eseguite a livello **locale** del singolo router. Ogni router **comunica** con il proprio routing component con gli altri per determinare la **propria routing table**.

- Logically - centralized control :

Approccio centralizzato in cui un **controllore** **logicamente centralizzato** calcola e distribuisce le routing table.

ALGORITMI DI ROUTING >

Gli algoritmi di routing del control devono minimizzare i costi, che sono dovuti a più fattori. [Come si calcolano i costi non è importante qui]

- NOTAZIONE: $G(N, E)$ G : Grafo N : Nodes E : Edges

$(x, y) \in G \leftrightarrow$ Arco connette x e y su G .

$(x, y) = (y, x)$: Bidirezionale • Di default costo $+\infty$

- CLASSIFICAZIONE:

- CENTRALIZZATO: Algoritmi consci globalmente di tutti i nodi e (o link state) i loro costi **prima** della loro esecuzione.

- DE-CENTRALIZZATO: Algoritmi iterativi che scoprono i nodi e i loro (o distance vector) costi **durante** l'esecuzione partendo da 1 nodo.
 → congestione

Altro criterio, oltre che siano o meno load sensitive:

- STATICO: Si basano sul presupposto che i nodi cambino **raramente**

- dinamico: Si basano sulla topologia e il traffico di rete, che causano cambiamenti frequenti.

ALGO. LINK STATE > Dijkstra

- POPOLAMENTO DATI

Tramite link state broadcast, dei pacchetti che vengono fatti inviare dal nodo a tutti i suoi nodi adiacenti contenenti inform. sui costi.

• ALGORITMO :

- SINTASSI: Dato calcolo da o a d

$\rightarrow N'$: Nodi in cui il percorso minimo è noto

$\rightarrow p(d)$: È il predecessore del cammino a costo minimo

$\rightarrow D(d)$: È il costo minimo fino a d corrente

1. Inizializzazione nodi:

$N' = \{o\}$ # Init

$\forall n \in N$ loop: # $N = \text{tutti i nodi}$

$D(n) = \text{se } \text{adj}(o, n) : c(o, n) \text{ else } +\infty$
End loop

Dato il nodo d'origine, \forall nodo n sono adiacenti la loro distanza minima è il costo, altrimenti $+\infty$

2. Ciclo:

while $N' \neq N$ do: # tutti i nodi sono in N'

$n \notin N' \mid D(n)$ sia minima

$N' = N' \cup \{n\}$

for $a \in \text{AdjList}(n) \mid a \notin N'$ then:

$D(a) = \min \{D(a), D(n) + c(n, a)\}$
end loop
end loop

L'algoritmo ha complessità logaritmica con le heap.

• OSCILLAZIONI

Variazioni continue in senso orario-antiorario preventi su reti con costi assimmetrici basati sul carico trasportato nei link. Solitamente sono risolte evitando che i router eseguano l'algoritmo nello stesso momento.

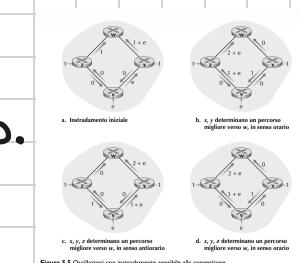
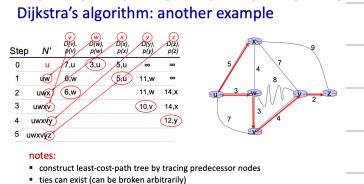
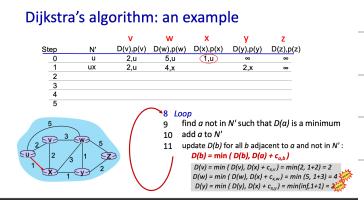
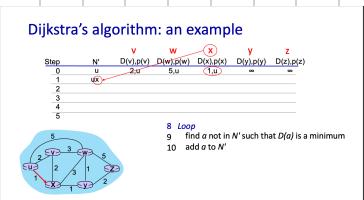
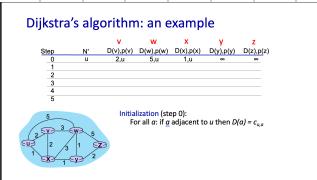


Figura 5.5 Oscillazioni con riconoscimento sensibile alle congestioni.

ALGO DISTANCE VECTOR > B.F.

- EQ. DI BELLMAN-FORD: $D_x(y) = \min_v C(x, v) + D_v(y)$

Eq. alla base dell'algoritmo di Bellman-Ford.

Describe che il costo min da x a y è il minimo fra tutti i vicini v a x su cui si calcola il min da v a y + costo $x \leftrightarrow v$.

• ALGORITMO: IDEA

Ogni nodo tiene con sé

$\rightarrow D_{st}$: Un vettore delle distanze stimate verso gli altri nodi.

$\rightarrow D_{vi}$: $=$ $=$ $=$ $=$ $=$ verso i suoi vicini.

Ogni tanto il nodo manderà D_{st} ai suoi vicini, che aggiorneranno la propria D_{st} con: $D_x(y) = \min_v C(x, v) + D_v(y)$ per ogni nodo y . Lo scambio farà convergere al costo min.

• ALGORITMO: PSEUDO-CODE

1. INIT: Nodo corrente = x

$\forall y \in N$ loop: ~~#~~ D_{st}

$$D_x(y) = \text{adj}(x, y) ? C(x, y) : +\infty$$

$\forall n \in \text{AdslList}(x)$:

$$D_n(y) = ?$$

[Inizio D_{st} a n]

2. CYCLO PRINCIPALE

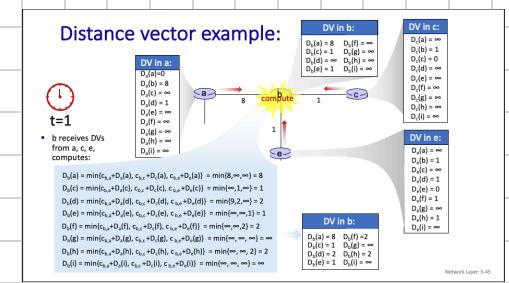
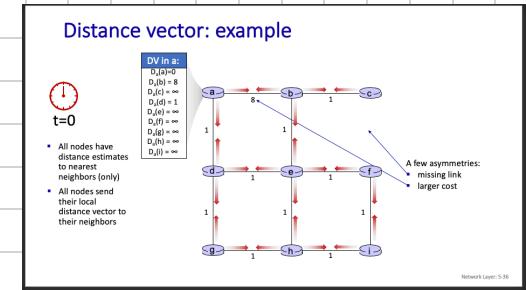
WAIT: o costo cambia V Ricevo vettore distanza.

$\forall y \in N$ loop:

$$D_x(y) = \min_v C(x, v) + D_v(y)$$

end loop

if D_{st} è cambiata: invia D_{st} ai vicini.



L21 >

-> **Algoritmo Bellman-Ford:**

- > Cosa succede quando i costi dei collegamenti cambiano in ribasso?
- > Cosa succede quando i costi dei collegamenti cambiano in rialzo?
- > Se ci sono dei problemi nei cambi dei costi, come si possono risolvere?
- > Elenca Complessità spaziale, Complessità temporale e Robustezza per:
 - > Algoritmi di Link State.
 - > Algoritmi di Distance Vector.

-> **AS:**

- > Quali sono le ragioni per cui non è possibile usare un LS o DV su tutta internet?
- > Cosa sono gli AS?
- > Da chi sono ergoati?
- > Come sono identificabili?
- > Gli ISP possono partizionarli?
- > Gestione del routing:
 - > Descrivi i tipi possibili di gestione del routing.
 - > Tutti i router possono usare lo stesso algoritmo?
 - > Cosa sono i gateway router? Questi quali tipi possibili di routing usano?

-> OSPF:

- > Che cos'è?
- > Su quale funzionalità, protocollo e algoritmo si basa?
- > Specifica quali sono le caratteristiche offerte da OSPF.

-> BGP:

- > Che cos'è?
- > Cosa fa? Quali sono le funzionalità che offre?
- > Path Advertisement:
 - > Che cos'è? come funziona?
 - > Su cosa si basa?
 - > Cosa fa un Router appena riceve un messaggio?
 - > Cosa sono le sessions BGP? Da cosa sono formate?
 - > Quali sono gli attributi dei messaggi BGP? Come sono valorizzati?
 - > Quali sono le tipologie di messaggi BGP? In quali casi vengono usate?
 - > In base a quali parametri viene filtrato il percorso di un pacchetto?

DISTANCE VECTOR > CAMBI DEI COSTI

A seconda di come cambiano i costi dei link, si possono avere comportamenti diversi dell'algoritmo

- COSTO DIMINUISCE:

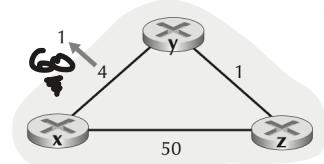
Le informazioni si propagano rapidamente.

- COSTO AUMENTA

Le informazioni si propagano lentamente, con comunicazioni potenzialmente infinite. (count to inf. problem)

- ESEMPIO: Con $\Delta_y(x) = 1$, $\Delta_z(x) = 5$
 $\Delta_y(z) = 4$, $\Delta_z(z) = 1$

t_1 : y calcola $\Delta_y(z) = \min\{60 + 0, 1 + 5\} = 6$ e notifica $\{\Delta_y(z)\}$
 t_2 : z riceve e $\Delta_z(z) = \min\{50 + 0, 1 + 6\} = 7$ e notifica $\{\Delta_z(z)\}$



Il costo verrà rimbalzato fra i z fino a che non raggiunge ∞ .

Il problema si può risolvere usando la poisoned reverse, che prevede di mentire con $\Delta(z) = \infty$.

ES: Nell'esempio, z dirà che $\Delta_z(x) = \infty$ fin quando z è instrada tramite y per raggiungere x .

- COPPARAZIONE

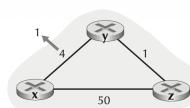
	Link State	Distance Vector
- Complexity	$O(n^2)$ mess.	Varia
- Speed	$O(n^2)$	Varia
- Robustness	Router avverte costo errato	Router avverte percorso errato

CONTROL PLANE > SCALABILITÀ & AUTONOMIA AUTORIZZATIVA

- Internet è formato da milioni di Host
- ISP vogliono poter decidere la gestione del forwarding nei router.

Per queste ragioni, non è possibile usare "solo" un singolo routing algo.

- AS: Autonomous Systems



e notifica
 t_0 : y nota change v
 t_1 : z riceve e aggiorna

Gruppi di router posti sotto lo stesso controllo amministrativo, partizionabili.

Sono erogati dall'ICANN e identificabili da un AS Number.

- GESTIONE DEL ROUTING

- INTRA-AS:

Routing interno all' AS. Tutti i router dello stesso AS **deveranno** seguire lo stesso intra-as algo.

- INTER-AS:

Routing fra AS. Sono eseguiti dai **gateway routers** (router con collegamenti con altri AS) che eseguiranno anche l'INTRA-AS.

I gateway routers avranno sia inter che intra AS entries nella routing table.

INTRA AS > Open Shortest Path First

È un protocollo intra-as di tipo **link state open source** che si basa sul flooding di pacchetti contenenti informazioni sullo stato dei collegamenti tramite protocollo IP (implementando RST en cap). Ottenute le informazioni, il singolo router **costruirà il grafo** ed eseguirà Dijkstra.

- CARATTERISTICHE

→ SICUREZZA: Configurabile per evitare manomissioni dei pacchetti scambiati. (Solo router fidati possono usare OSPF).

→ GESTIONE PESI : I pesi nei collegamenti sono configurabili dagli admin (stesso costo = percorso minimo).

→ SUPPORTO A MULTICAST FORWARDING

→ GESTIONE GERARCHICA

Un AS OSPF può essere diviso in varie aree con diverse versioni di OSPF (es. diversi pesi):

→ AREA BORDER ROUTER:

Router di confine collegato con un'area locale e il backbone

→ BACKBONE:

Area dorsale che effettua il routing verso le aree locali tramite le area border router.

INTER AS ➤ Broader Gateway Protocol

È il protocollo inter-as "de facto," implementato come distance-vector de-centralizzato e async.

• CARATTERISTICHE

In BGP, l'indirizzamento avviene verso un prefisso CIDR, rappresentante una subnet o una collezione di subnet. Ogni record nella Tabella avrà CIDR e porta routing. Questo consente di:

1. OTTENERE INFO: (PATH ADVERTISEMENT)

Sulla raggiungibilità del prefisso da parte del sistema confinante

2. CALCOLARE IL PERCORSO MINIMO: Verso il prefisso.

• PATH ADVERTISEMENT: FUNZIONAMENTO

Ogni coppia di router si scambia informazioni sui prefissi raggiungibili usando connessioni TCP semi-permanenti, costituendo una rete a maglia di connessioni fra tutti i router.

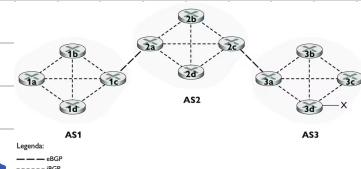
Alla ricezione di un messaggio, il router salva le informazioni sulla sua routing table e inoltre il messaggio verso tutti i router; i gateway router raggiunti provvederanno a inoltrare i messaggi verso gli AS confinanti, ripetendo gli step.

• SESSIONI BGP

Definisce la connessione TCP e i messaggi scambiati riguardo un prefisso specifico.

- i BGP: Coinvolge 2 router dello stesso AS.

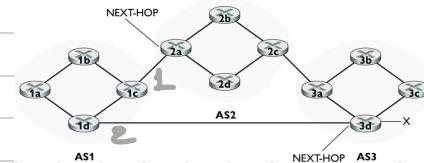
- e BGP: // == di AS diversi.



• ATTRIBUTI:

Ogni advertisement scambiato in una sessione BGP viene arricchito con attributi:

- AS-PATH: 1: "AS2 AS3" 2: "AS3"



Contiene il percorso (come lista di ASN) attraversato dal messaggio, organizzati dal più vicino al più lontano.

- NEXT HOP:

Contiene l'interfaccia del router del 1° AS nell'AS-PATH, questa sarà quindi l'IP esterno del gateway esterno collegato.

Ex: 1: 2a (porta con 1c); 2: 3d (porta con 2d).

• TIPOLOGIE DI MESSAGGI:

- OPEN: Inizializza la connessione TCP.
- KEEP ALIVE: Mantiene la connessione attiva o fa Ack di OPEN.
o aggiunge
- UPDATE: Aggiorna le info. di raggiungibilità del prefisso
- NOTIFICATION: Per errori o chiusura di connessione.

• ROUTING BGP:

Dati più percorsi disponibili, si filtra il percorso in base a

1. PREFERENZA LOCALE: Espressa tramite pesi impostati dall'admin dell'AS sui router.

2. AS PATH PIÙ PICCOLO:

3. NEXT-HOP PIÙ VICINO / HOT POOL ROUTING:

Quanto, il router userà l'intra-as routing per verificare i costi per arrivare dei next hop, per poi scegliere il minore e trovare l'interfaccia V.

4. ALTRI ATTRIBUTI BGP.

L22 >

-> **Link Layer:** Intro

- > Quali sono le responsabilità del link layer?
- > Comunica attraverso un protocollo reliable?
- > Quali sono gli elementi fondamentali del link layer?
- > Quali sono le sue caratteristiche principali?
- > Le funzionalità del link layer sono implementate in hardware o software?

-> **Link Layer:** Rilevazione degli errori

- > Come funziona a livello concettuale? È garantita l'assenza di errori? Cosa necessita?
- > Parity Check:
 - > Come funziona? Come viene rilevato un'errore?
 - > Quali sono i suoi vantaggi/svantaggi? Ci sono varianti?

-> Checksum:

- > Come funziona?

-> CRC:

- > Come funziona? Come viene rilevato un'errore?
- > Quali sono le formule di cui si serve?
- > Quanti errori consecutivi è in grado di sopportare?

-> **Protocolli ad accesso multiplo**:

- > Quali sono le tipologie di collegamenti possibili?
- > Quali sono gli obiettivi dei protocolli ad accesso multiplo?
- > Qual'è lo scenario ideale di un protocollo ad accesso multiplo? Quali sono i suoi requisiti?
- > Channel Partitioning:
 - > Elenca le tipologie di protocolli elencando vantaggi e svantaggi.
 - > Elenca le tipologie di protocolli elencando vantaggi e svantaggi.

-> Random Access Protocols:

- > Su che idea si basa?

-> Slotted ALOHA:

- > Su che premesse si basa?
- > Come funziona?
- > Cosa succede se non si rilevano collisioni? Se si rilevano invece?
- > Elenca le formule di cui si serve per determinarne il successo.
- > Quali sono i vantaggi e svantaggi? È efficiente per tanti nodi comunicanti?

-> Pure ALOHA:

- > Su cosa si basa?
- > Qual'è la probabilità che un nodo trasmetta?
- > È efficiente per tanti nodi comunicanti? Con che valore si stima la sua efficienza?

-> CSMA:

- > Come funziona la versione base?
- > Come funziona CSMA/CD?
 - > Cosa fa se il canale non è libero?
 - > Cosa fa se succedono collisioni?
 - > È in grado di rilevare le collisioni dopo la trasmissione?
 - > Come viene calcolata la sua efficienza?

LINK LAYER > INI

Il link layer ha la responsabilità del trasporto dei datagrammi fra nodi adiacenti collegati da un link tramite un protocollo di link reliable o meno.

• DICTIONARY

- Nodo: Qualsiasi dispositivo che opera nel link layer
- Link: Comm. link che collega nodi adiacenti.

• CARATTERISTICHE:

→ FRAMING: Incapsulamento in frame effettuato da quasi tutti i protocolli link.

→ LINK ACCESS:

Il protocollo Medium Access Control stabilisce le regole con cui inserire il frame nel collegamento.

→ RELIABLE DELIVERY:

Simile all'ROT TCP su comm. link con alto tasso d'errore (Wi-Fi).

→ ERROR DETECTION & CORRECTION: Tramite frame datagram agg.

• IMPLEMENTAZIONE LINK LAYER

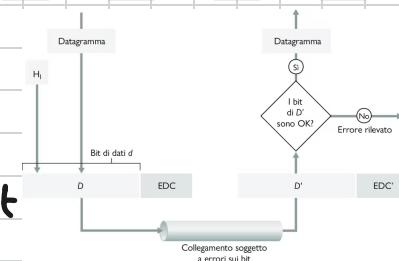
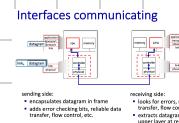
È implementato nella Network Interface Card (HW) all'interno di ogni host. All'interno della NIC, è presente un controller in HW un chip dedicato che implementa la maggior parte delle funzioni. Le funzioni di alto livello sono implementate in SW.

LINK LAYER > RILEVAZIONE ERRORE

• LIVELLO CONCETTUALE

Il sender invia i dati D e aggiunge dei bit di Error Detection and Correction.

Il receiver riceve dal comm. link D' , EDC' dal sender e



controllera' la presenza di errori.

Non viene garantita l'assenza di errori, ma si cerca di minimizzarne la probabilità con alta ridondanza di EDC bit.

- PARITY CHECK

Prevede di aggiungere 1 bit in modo che il # di 1 nei bit del datagramma sia pari/dispari a seconda dello schema usato.

- 2D PARITY BIT

Variante a 2 dimensioni con righe e colonne di parity bits creata per poter rilevare precisamente gli errori e correggerli.

- CHECKSUM: Solitamente non usata

- CRC: CYCLIC REDUNDANCY CHECK

Prevede che sender e receiver si accordino su una stringa di $r+1$ bit, detta Generatore, la quale avrà 1 come bit più significativo.

Il sender, dovrà poi scegliere r bit di EDC, R , da appendere ai d bit dei dati D . L'idea prevede che la stringa $D+R$ interpretata come numero binario sia esattamente divisibile per G (in binario, con resto \emptyset). Se il resto $\neq \emptyset$, c'è un'errore.

- FORMULE:

$\rightarrow D \cdot 2^r \text{ XOR } R$: Definisce la stringa di r bit divisibile da G

$\rightarrow (D \cdot 2^r / G) \text{ resto}$: Il resto è R .

CRC puo' rilevare fino a $r+1$ errori consecutivi.

PROTOCOLLI DI ACCESSO MULTIPLO >

I collegamenti possono essere

- POINT-TO-POINT: Non richiedono particolari protocolli di accesso.

Senza errori	Errore correggibile di un singolo bit
1 0 1 0 1 1	1 0 1 0 1 1
1 1 1 1 0 0	1 0 1 1 0 0
0 1 1 1 0 1	0 1 1 0 1 1
0 0 1 0 1 0	0 0 1 0 1 0

Figura 6.5 Parità pari bidimensionale.

- BROADCAST:

Prevedono la presenza di più nodi trasmettitori e ricevitori nel link, che potrebbero provare a trasmettere simultaneamente.

I protocolli di accesso multiplo puntano a minimizzare le possibilità di collisione (trasmissioni simultanee che corrompono i frame) e ad efficientare l'uso del canale.

• SCENARIO IDEALE:

Dato un canale broadcast con velocità R b/s:

1. Se 1 nodo trasmette : Deve avere throughput R b/s

2. $\approx N$ nodi trasmettono: $\parallel \quad \parallel \quad \parallel = R/N$ b/s.

3. Deve essere decentralizzato (non esistono nodi master).

4. Deve essere semplice.

• CHANNEL PARTITIONING

Incluse tecniche come:

• Time Divided Multiplexing:

Divisione del tempo in timeslot assegnati singolarmente ai nodi

+ rispetta le regole dello scenario ideale

- inefficace nel caso di un singolo sender e tanti receiver per i tempi morti e il throughput limitato a R/N .

• Frequency Divided Multiplexing:

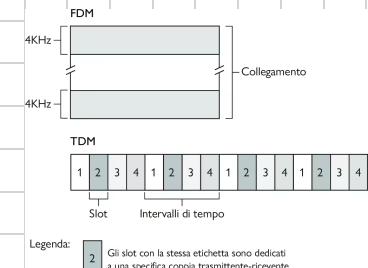
Comunicazione in canali di frequenza riservati per sender.

+ rispetta lo scenario ideale

+ non è limitata dal tempo.

- È limitato dal throughput R/N anche con pochi sender e tanti receiver.

• Code Divided Multiplexing:



Senders e receivers comunicano codificando con un codice assegnato, decodificato dai receivers
+ consente trasmissioni simultanee.

- RANDOM ACCESS PROTOCOLS

Tutti i sender trasmettono alla massima velocità del canale.
Se si verifica una collisione, verranno ritrasmessi i dati dopo un periodo casuale.

- SLOTTED ALOHA:

Protocollo RAP che ha come premessa

- pacchetti da L bits.
- time slots da L/R b/s (di trasm.).
- nodi in sync sui time slots che rilevano le collisioni prima del termine del timeslot.

Il funzionamento prevede che un nodo che deve trasmettere aspetta l'inizio del time slot successivo:

- se non vengono rilevate collisioni: la trasmissione è terminata bene.
- se collisioni vengono rilevate: proverà a ritrasmettere con probabilità p nelle trasmissioni successive fino a che non riesce.

Le probabilità generali sono, dati N sender:

- $p(1-p)^{N-1}$: Probabilità che un dato nodo abbia successo.
- $N \cdot p(1-p)^{N-1}$: $\approx \approx \approx$ qualsiasi $\approx \approx \approx$.

+ È semplice e decentralizzato

\hookrightarrow Efficienza

+ È efficace con pochi sender simultanei

- Nel caso peggiore, solo il 37% delle trasmissioni va a buon fine

- PURE ALOHA

Versione di ALOHA senza timeslot. Quando un frame da

trasmettere arriva, il nodo lo trasmette subito.

$\frac{1}{2}(1-p)$

- $p \cdot (1-p)$: probabilità di trasmissione di un nodo (efficienza)

+ Completamente decentralizzato

- 1/8r di trasmissioni a buon fine nel caso peggiore (metà r slotfull)

• CSMA:

Protocollo RAP disponibile in 2 versioni

- CSMA :

Verifica che non vi siano altre trasmissioni in corso prima di trasmettere.

In caso di collisioni: continua la trasmissione

- CSMA with Collision Detection (CSMACD)

Versione che integra il rilevamento delle collisioni

1. La NIC verifica se il canale è libero:

No: Resta in attesa che si liberi

Sì: Procede alla trasmissione:

? Se vengono rilevate collisioni durante la trasmissione

1. Firma la trasmissione

2. Aspetta del tempo casuale, in Ethernet è usato

l'Exponential backoff:

a. Si sceglie K da $\{0, 1, 2, \dots, 2^{n-1} - 1\}$

con $n = \#$ di collisioni, con n massimo 10.

b: Aspetta il tempo di trasmissione di $K \cdot 512 \text{ b}$

trasmissioni

Per quanto riguarda l'efficienza (tempo con \vee senza collisioni)

$$EFF = \frac{1}{1 + S \cdot d_{prop} / d_{transm}}$$

È alta per: - $d_{prop} \rightarrow \infty$
- $d_{transm} \rightarrow +\infty$

L23 >

-> **Protocolli ad accesso multiplo:** Taking Turns

-> Quali erano le caratteristiche che non rispettavano i protocolli RAP?

-> Descrivi tutti i protocolli, spiegando vantaggi e svantaggi.

-> **MAC Address:**

-> Che cos'è?

-> A cosa serve?

-> Da chi sono emessi?

-> Come sono formati?

-> Come viene relazionato il MAC Address nell'invio e ricezione di frame?

-> Gli switch hanno MAC address? Perché?

-> **ARP:**

-> Cos'è?

-> ARP Tables:

-> Cosa sono?

-> Dove sono salvate?

-> Qual'è la loro struttura?

-> Spiega come ARP viene usato nelle comunicazioni della stessa subnet.

-> La funzionalità principale di ARP puo' essere usata anche per host su subnet diverse?

-> Spiega se e come ARP viene usato nelle comunicazioni su subnet diverse.

-> Descrivi la struttura di un pacchetto ARP.

-> **Ethernet:**

-> Che cos'è?

-> Descrivi i campi che formano un frame eth.

-> Descrivi le caratteristiche di Ethernet

-> **Switch:**

-> Che cos'è?

-> Quali sono le sue funzionalità principali?

-> Switch Table:

-> Che cos'è?

-> Che struttura presenta?

-> Come viene usata?

-> Descrivi i casi principali con cui viene usato.

-> I record della switch table hanno una data di scadenza?

Sono protocolli ad accesso multiplo che soddisfano il 2° requisito (Throughput R/N per N nodi) che non veniva rispettato dai RAP.

- POLLING ES: Bluetooth

Usa un nodo centrale che orchestra la comunicazione in maniera circolare (es: Prima 2, poi 3,...). Il nodo centrale capisce che è finito il turno di un nodo dall'assenza di segnale. Poi notifica al prossimo nodo che è il suo turno.

+ Riduce i tempi morti

- Throughput inferiore a R/N per il tempo di notifica.
- Se il nodo centrale si guasta, non funziona più.

- TOKEN PASSING

Usa un frame centrale detto token che viene passato da tutti i nodi in modo circolare. Se il nodo non trasmette lo passa subito.

+ Decentralizzato

- Se il nodo si rompe non funziona più
- Richiede retry policy se l'invio fallisce

INDIRIZZO MAC >

È un'indirizzo a liv. collegamento che identifica **univocamente** una N.I.C. di un nodo e consente il trasporto dei frame nel link.

- STRUTTURA

È formato da 6 byte con **struttura piatta** (senza gerarchie) che non cambiano mai. Sono emessi dalla IEE alle aziende.

- FUNZIONAMENTO

- INVIO: La NIC aggiunge il MAC al frame e spedisce.
- RICEZIONE:

Si verifica il MAC Address di destinazione del frame arrivato:

- È per il nodo → Genera interrupt lato network.
- È in broadcast (FF-FF..;FF) → .
- Altrimenti: → scarta

NB: Gli switch non hanno mac address perché fanno solo fwd

ADDRESS RESOLUTION PROTOCOL >

Protocollo fra liv. Link e Network che traduce gli ind. IP in MAC

Sfrutta le ARP Tables, tabelle plug & play salvate in ogni nodo per poter comunicare.

- STRUTTURA ARP TABLE: <IP, MAC, TTL>

Indirizzo IP	Indirizzo MAC	TTL
222.222.222.221	88-B2-2F-54-1A-0F	13:45:00
222.222.222.223	5C-66-AB-90-75-B1	13:52:00

- FUNZIONAMENTO: STESSA SOTTORETE

Il sender invia un datagramma. Nel link layer si controlla se una entry esiste nella ARP Table

V. ESISTE: Parte il frame

F. NON ESISTE:

1. Il sender invia una ARP Request in broadcast specificando l'IP del receiver.

2. - Il receiver con l'IP desiderato risponderà solo al sender con il suo MAC.

- Gli altri receivers potranno salvare il MAC del sender.

3. Il sender aggiorna la sua ARP Table con un TTL (20 min).

NB: ARP può risolvere gli indirizzi solo della stessa subnet

- FUNZIONAMENTO: FRA SUBNET DIVERSE:

Il nodo invierà il frame all'interfaccia del router, che eventualmente raggiungerà l'interfaccia corretta e imposterà il MAC di destinazione.

Routing to another subnet: addressing

- A creates IP datagram with IP source A, destination B
- R creates link-layer frame containing A-to-B IP datagram

R's MAC address is frame's destination

MAC src: 7A-2B-8C-0D-0E-0F

MAC dst: 1A-23-F9-C0-06-00

IP src: 111.111.111.111

IP dst: 111.111.111.112

TTL: 255

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

IP header: 111.111.111.111 -> 111.111.111.112

IP payload: 111.111.111.111 -> 111.111.111.112

LCP header: 0x00 0x00 0x00 0x00

LCP payload: 0x00 0x00 0x00 0x00

</

• STRUTTURA ARP 9 Campi:

1. HW Type 2. Protocol Type : (Net Layer)

3. HW Length: Lunghezza in bytes del phy. Add.

4. Protocol // : // dell' indirizzo logico.

5. Operation: 1: Request, 2: Reply

6-7 Sender / Target HW Address: Indirizzo fisico

8-9 // // Protocol // : // logico

ETHERNET >

Protocollo popolare per la trasmissione di dati cablati.

• ETH FRAME STRUCTURE : 6 Campi

1. Preamble: Usato per sync. i clock di sender (receiver 7 bytes) e risvegliare le schede. (10101010 10101011)

2-3 Dest/Source Address: Indirizzi MAC. (6 bytes)

4. Type (2 byte): Per definire il tipo protocollo (ES: 8006 ARP)

5. Data: Dati. Dim. da 46-1500 bytes MTU.

6. CRC: Per EDC.

• CARATTERISTICHE:

→ No handshake
 → Connection-less → Unreliable → CSMA/CD con backoff

ETH SWITCH >

Dispositivo di link che fa filtering (scarto o meno) e forward

• FORWARDING:

Usa una switch table contenente:

- MAC di destinazione
- Interfaccia dello switch collegata a tale nodo
- Data di creazione

Hardware length	Protocol length	Operation	Protocol type
Sender hardware address (For example, 6 bytes for Ethernet)		Request 1, Reply 2	
Sender protocol address (For example, 4 bytes for IP)			
Target hardware address (For example, 4 bytes for Ethernet) (It is not filled in a request)			
Target protocol address (For example, 4 bytes for IP)			

Alla ricezione di un frame lo switch controllera' se:

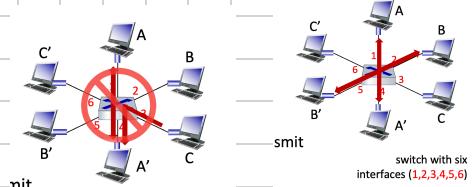
- frame dest non switch table \rightarrow broadcast a tutti (no render)
- // // a se stesso \rightarrow scarto
- // // a dest in switch table \rightarrow forward al dest.

• AUTO - APPREND:

La switch table viene popolata in maniera intelligente ad
ogni invio. Dopo un aging time senza invio, il record ^{viene} ~~ha~~ rimossa.

• CONN. SIC.

Ogni link è a sé stante. Coll.
ridotte



L24 >

-> **Wireless:** Introduzione

-> Elementi di Rete:

- > Che cos'è un wireless host?
- > Che cos'è un wireless link?
- > Che cos'è una base station?
 - > Con cosa comunica?
 - > A cosa è collegata?
 - > Che cos'è l'handoff?

-> Modalità di connessione:

- > Definisci le modalità.
- > Entrambe si collegano ad una base station?
- > Come interagiscono con la comunicazione con altri host?

-> Tassonomia: Definisci le combinazioni possibili

-> Definisci le differenze che caratterizzano il wireless da phisycal link

-> **Wireless:** Misurazioni

-> SNR:

- > Che cos'è?
- > Come si misura?
- > Cosa succede quando SNR è grande?

-> BER:

- > Che cos'è / cosa indica?
- > Da cosa sono legati SNR e BER?
- > Cosa succede quando si fissa una tecnica di modulazione e si incrementa il BER?
- > Cosa succede quando si fissa un BER e si "incrementa" la tecnica di modulazione?
- > Definisci gli scenari in cui la tecnica di modulazione cambia, se cambia.

-> Definisci gli altri problemi che caratterizzano la comunicazione wireless.

-> **Wireless:** CDMA:

- > Che cos'è?
- > Descrivi il suo funzionamento.

-> **WiFi:** Intro

- > Descrivi l'architettura su cui si basa il WiFi.
- > Quali sono le informazioni necessarie per configurare un AP? Ci sono criticita?

-> Scanning:

- > Come fanno gli AP a rendersi visibili verso l'esterno?
- > Che informazioni vengono inviate?
- > Come puo' essere classificato lo scanning? Quali sono le differenze (se ci sono)?
- > Cosa succede una volta selezionato l'AP?

-> **WiFi:** CSMA

- > Che cos'è e quali sono le caratteristiche di CSMA?
- > Descrivi l'algoritmo CSMA

WIRELESS INTRO >

• ELEMENTI DI RETE

- WIRELESS HOST: Equivalente all'host fisico.

- WIRELESS LINK:

Link usato dal wireless host per comunicare con Base Station o wireless host.

- BASE STATION:

Elemento di infrastruttura che invia/riceve pacchetti dal/per il wireless host associato, con altri wireless nodes o con nodi cablati. Solitamente è collegata al network wireless.

- HANDOFF: passaggio da 1 base station all'altra.

• MODALITÀ DI CONNESSIONE

- INFRA. NODE:

Connessione tramite base station, che fa da tramite ai servizi di rete (cablata) offerti.

- AD-HOC NODE:

Connessione senza base station (infra). Per comunicare ci sono bisogno di implementare servizi come routing, addressing...

• COMBINAZIONI POSSIBILI (TASSONOMIA)

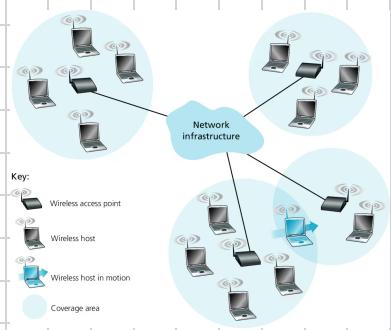
	SALTI RETE	1 Salto	> 1 Salto
INFRA			
SI	(WLAN, WiFi)	1 Coll. a Internet	(mesh wireless) Più salti con nodi wireless pre-Internet
NO	Host Coordina altri (Bluetooth)	Collegam. hoc (MANET, VANET)	

DIFFERENZE CON LINK FISICO >

• SIGNAL ATTENUATION:

Per gli ostacoli attraversati e la distanza.

• INTERFERENCE: Da comunicazioni o rumori elettromagnetici



altre in 2.4 GHz
alla stessa frequenza (microwave).

- MULTIPATH PROPAGATION:

Tempi di arrivo variabili dovuti a rimbalzi del segnale con altri elementi fisici che causano disturbo.

WIRELESS > MISURAZIONI

- SIGNAL-TO-NOISE-RATIO (SNR)

Misura in decibel dB dell'intensità del segnale ricevuto.

SNR Grande \rightarrow Sarà più facile distinguere segnale e rumore.

- BIT ERROR RATE (BER)

Indica la probabilità che i bit inviati siano ricevuti con errore.

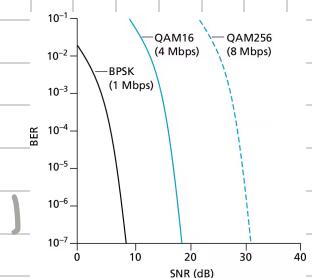
SNR e BER sono legati dalla tecnica di modulazione del segnale usata.

- DATO T. MODULAZIONE: (BPSK da 10^{-2} a 10^{-7})

incrementando l'SNR si riduce il BER.

- DATO SNR: Per SNR 10dB: BPSK (10^{-2}) \rightarrow QAM16 (10^{-5})

"incrementando" T. di modulazione (in Ny), aumenta il BER

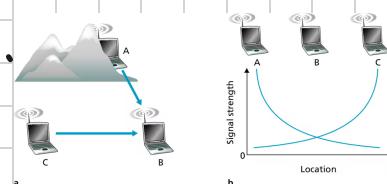


Le T. modulazione cambiano con la **mobilità** e per adattarsi meglio al layer fisico.

WIRELESS > ALTRI PROBLEMI

- HIDDEN TERMINAL PROBLEM: (a)

Collisioni causate da nodi che potrebbero ma non possono vedersi per la presenza di ostacoli fra i 2.



- FADING: (b)

Collisioni avvenute nel receiver in cui i sender non riescono a rilevare il segnale dell'altro.

WIRELESS > CDMA

È un protocollo ad accesso multiplo di tipo channel partitioning

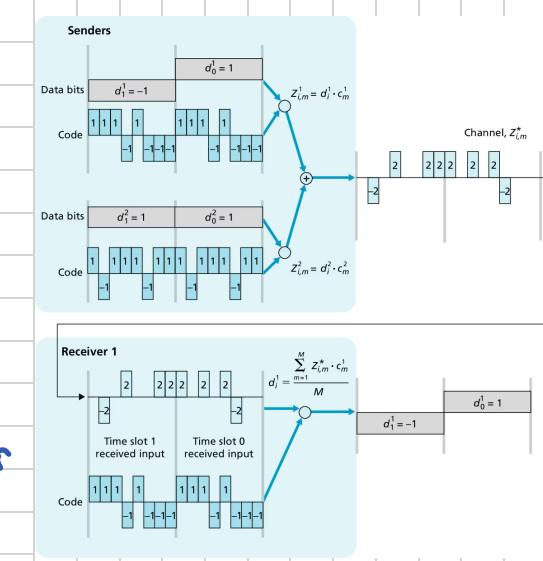
usato nelle reti wireless.

• FUNZIONAMENTO:

Dati:

- d_i^s : Il bit i-esimo da trasmettere dal sender s-esimo

- c_m : Il bit m-esimo del codice usato per codificare il segnale



• TRASMISSIONE

Il segnale trasmesso avrà forma bit per bit come.

$$Z_{i,m}^s = d_i^s \cdot c_m$$

Alla posizione m del segnale i per s sender.

Verrà sommata al segnale degli altri sender.

• DECODIFICA:

Ogni sender decodificherà bit-per-bit dei dati come:

$$d_i^s = \frac{1}{M} \sum_{m=1}^M [Z_{i,m}^s \cdot c_m]$$

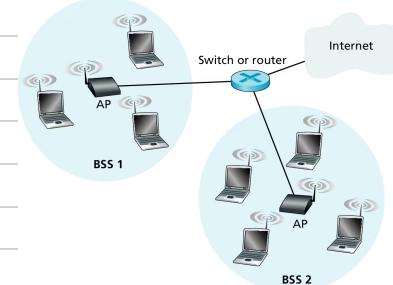
WIFI \rightarrow standard IEEE 802.11

• ARCHITETTURA:

• BSS : BASIC SERVICE SET :

Contiene 1 o più wireless stations (Host) e 1 base station (detto Access Point).

Ogni wireless e base stations ha un MAC address.



• CONFIGURAZIONE

Ogni access point necessita della configurazione di:

- SSID : Service Set ID

- Canale di frequenza:

Può essere lo stesso di altri service set, in quanto su 11 canali, un canale non è sovrapposto se ci sono almeno 4 canali.

di distanza (802.11)

● SCANNING:

Wi-Fi impone che gli AP inviano dei beacon frame contenenti SSID e MAC address.

Le wireless station possono associarsi in maniera:

- PASSIVA: Ascoltando i beacon frame degli AP.
- ATTIVA: Scambio di probe (sonde) request/response (frame) fra host e AP contenenti il range dell'host.

Selezionato l'AP, si associa tramite lo scambio di association request/response (frame).

Wi-Fi > MEDIUM ACCESS CONTROL

Usa il CSMA con Collision Avoidance che:

- termina l'invio dei pacchetti una volta iniziata la trasmissione
- non prevede collision detection.
- usa un sistema di acknowledgement dei pacchetti.

● ALGORITMO

- SENDER:

1. Verifica se il canale è occupato per tempo DIFS

a. libero:

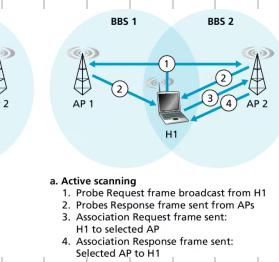
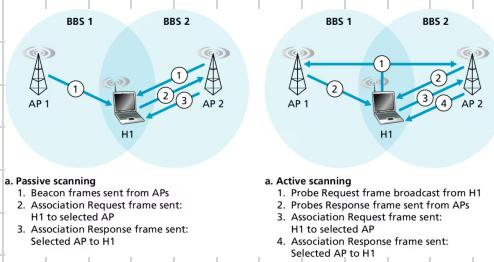
b occupato:
1. Genera k da un algoritmo di exp. backoff (CSMA CD)

2. Decrementa k solo quando il canale è libero

3. inizia la trasmissione del frame

a. Ack arriva: continua dal punto 1 per altri frame

b. Ack non arriva: ri-trasmetti.



- RECEIVER:

1. Riceve e verifica se il frame è privo di errori (CRC):

a. OK: Invia Ack dopo tempo SIFS.

L25 >

-> **WiFi:**

-> RTS/CTS:

- > Cosa si intende per RTS/CTS?
- > In che modalità vengono inviati i frame?
- > Cosa deve specificare un host/AP in RTS?
- > Cosa deve specificare un host/AP in CTS?
- > Viene usato sempre?

-> Struttura Frame:

- > Descrivi tutti i campi della struttura di un frame Wi-Fi.
- > Come interagisce (se lo fa) l'AP con i frame Wi-Fi ed Ethernet?

-> Mobility: Stessa subnet

- > La mobility nella stessa subnet è applicabile facilmente?
- > Vengono mantenute le connessioni TCP?
- > Quali elementi infrastrutturali di rete vengono coinvolti in questo processo? Come?

-> Rate adaption:

- > Che cos'è?
- > Quando viene applicata?
- > Quali sono i fattori che ne influenzano l'applicazione?
- > Ci sono similitudini con altri protocolli/funzionalità di altri protocolli?

-> Power management:

- > Come si comportano wireless station ed AP nella gestione energetica?
- > Come viene usato il beacon frame in questo contesto?

-> **Bluetooth**:

- > Come può essere classificata?
- > Quali sono le sue caratteristiche?
- > Come viene organizzata?

-> **Mobile**: 4G/LTE:

- > Come vengono suddivise le aree geografiche?
- > Descrivi dettigliatamente gli attori del mondo mobile.
- > Come viene identificato un host nel mondo mobile?
- > Dove vengono custodite le sue informazioni principali?
- > Quali sono le funzionalità offerte dalla base station?
- > Quali sono i servizi del control plane? Di cosa sono incaricati?
- > Come viene gestito il percorso fra l'host e i gateway?

-> Gestione layer:

- > Quale layer è gestito in maniera speciale dalle reti mobili? In che modo è gestito?

-> Schemi di comunicazione:

- > Come funziona la comunicazione da host a base station?
- > Come funziona la comunicazione da base station a gateway?
- > Quali sono i protocolli usati nella comunicazione? In che contesto?

-> 5G:

- > È retro-compatibile?
- > Su cosa si basa?
- > Quali sono i suoi vantaggi e svantaggi?

Funzionalità di collision avoidance basata su prenotazione

L'idea si basa sull'invio in broadcast dei frame:

- RTS: (Req. To Send)

Usato per richiedere ad AP il permesso di inviare dati.

Questa specifica il **tempo totale richiesto** per l'invio dei dati e ACK.

- CTS: (Clear-To-Send)

Eventuale risposta dell'AP specificando il SLAC che ha il perm.

Gli altri nodi eviteranno di comunicare per il tempo specificato.

Ciò risolve l'**hidden terminal problem**. Per le collisioni, è ancora possibile che ci siano ma saranno di breve durata.

- DRAWBACKS & USO

Siccome l'uso di RTS/CTS aggiunge overhead, questo viene abilitato per dati di grandi dimensioni (> frame).

FRAME 802.11 > STRUTTURA

- CRC: 4 Byte di CRC per rilevamento di errori

- PAYLOAD: Contiene datagramma IP o pacchetto ARP.

- INDIRIZZI 1, 2, 3 e 4:

Indirizzi MAC usati nella comunicazione:

- MAC 1: Indirizzo del receiver.

[Wireless ST (AP)]

- MAC 2: // sender.

[AP/ wireless st]

- MAC 3: // dell'interfaccia del router connesso all'AP.

- MAC 4: Per le reti ad-hoc.

In infrastructure mode, AP e router modificano i frame:

- DA ROUTER A WIRELESS HOST:

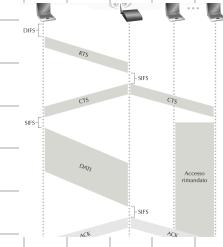


Figura 7.13 Struttura di un frame 802.11.

L'AP crea il frame wi-fi e imposta: NODO, AP, ROUTER come MAC.

- DA HOST A ROUTER

1
det
src
2
sic
dest
3
route

L'AP crea il frame ethernet e imposta: NODO, ROUTER come MAC.

- DURATA: Usata dallo schema RTS/CTS
- SEQ. NUMBER: Come nell' RDT del Transport Protocol, consente di distinguere nuove trasmissioni / ri-trasmissioni.

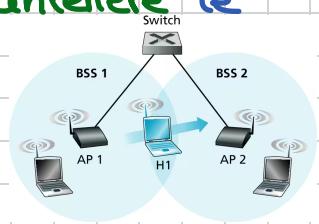
- FRAME CONTROL:

- Type / Subtype: Identifica RTS/CTS / DATA/ACK
- To-AP / From AP: Definisce il significato dei 4 indirizzi mac.

(se l'AP sta inviando o meno il pacchetto).

MOBILITY > FRA AP DELLA STESSA SUBNET

Quando una wireless station si muove puo' mantenere le stesse connessioni (TCP) aperte.



Questa situazione si verifica quando gli AP sono connessi ad un dispositivo non router (switch).

L'idea è di sfruttare l'auto-apprendimento dello switch tramite l'invio di un frame broadcast (switch non hanno mac) per triggerare l'aggiornamento delle switch tables.

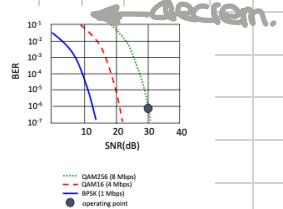
FUNZIONALITA' AVANZATE 802.11 >

- RATE ADAPTATION: Simile a Congestion Control di Transport

Cambio dinamico della tecnica di modulazione per far "aumentare" il transmission rate a seconda del SNR:

→ Incremento: Se si ricevono 10 ACK di fila o se il timer dall'ultimo decremento scade.

→ Decremento : Se non arrivano ACK, ovvero il BER diventa troppo alto



• POWER MANAGEMENT:

Wireless station e AP efficientizzano la loro gestione

→ Wireless Host:

Informa l'AP nei frame trasmessi che sta per entrare in modalità sleep e mette un timer per svegliarsi prima che arrivi il prossimo beacon frame.

Il beacon frame conterrà una eventuale lista di frame da recapitare all'host, che potrà richiederli con un polling frame

→ AP: Effettuerà il buffering dei frame destinati all'host.

BLUETOOTH > Personal Area Network

È una PAN/piconet ad-hoc a basso range (10 m) e bassi consumi.

• ORGANIZZAZIONE

Usa un sistema di **polling**, con un nodo master che determina power (energia), a quale device i sender trasmettono e i time-frame su cui si basa il MAC (TDM).

MOBILE > 4G / LTE

• ARCHITETTURA

La rete cellulare è suddivisa in aree geografiche dette **celle**.

Ogni cella ha **1 base station** che ne gestisce la connettività.

Gli attori mobile sono:

• USER EQUIPMENT : Telefono, Tablet, Sensori,...

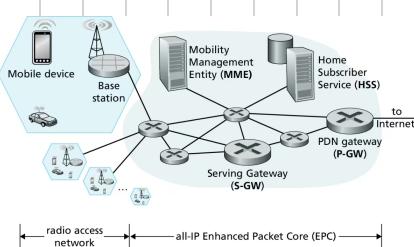
Definisce qualsiasi dispositivo finale che

invia/riceve informazioni nella rete cellulare (Host).

Ogni UE dispone di una SIM, la quale fornisce:

- INTERNATIONAL MOBILE SUBSCRIBER IDENTITY: (IMSI):

64 bit che identificano l'U.E. in tutta la rete cellulare (o MAC)



- Info sui servizi assegnati all'utente (abbonamenti).
- chiavi di cifratura usate per l'authN (vedi sicurezza).

- eNODEB :

È la base station, che nelle reti mobili si occupa anche di:

- MOBILITY: Tramite la coordinazione con le altre base station.
- EFFICIENTAMENTO: Delle frequenze usate
- TUNNELLING: Per la comunicazione fra U.E. ed i gateway.

- HOME SUBSCRIBER SERVICE (H.S.S.):

Database del **control plane** che storica i dati degli U.E.

che hanno la rete dell'H.S.S. come **home network**.

È usato con M.M.E. per l'**autenticazione**.

- SERVING GATEWAY (S-GW) /

- PACKET DATA GATEWAY (P-GW)

Sono router che **mentono** sul datapath intrapreso dal mobile device al resto della rete (oscurano H.S.S. ed M.M.E.).

- P-GW

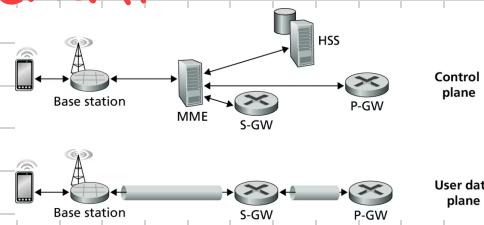
Il P.-GW è anche un'astrazione verso l'esterno in quanto ultimo router prima del resto della rete.

- MOBILITY MANAGEMENT ENTITY (M.M.E)

Elemento di **control plane** focalizzato su:

- TUNNELLING: Come supporto a S-GW e P-GW.
- LOCATION: Tiene traccia della posizione dell'U.E. man mano che si sposta fra celle grazie agli eNodeB.
- AUTHENTICATION:

Dialoga con l' H.S.S. della home network da cui provengono i Foreign U.E. per verificarne l'auth. (Vedi sicurezza).



● ARCHITETTURA: LINK LAYER

Il link layer mobile è diviso in 3 categorie:

- PACKET DATA CONVERGENCE:

Applica

- compressione per ridurre la dimensione dei bit trasmessi.
- crittografia tramite le chiavi negoziate con R.M.E.

- RADIO LINK CONTROL:

Effettua frammentazione ed RDT (simile ad algo. CSMA/CA).

- MEDIUM ACCESS CONTROL: Richiesta ed uso degli slot radio.

● SCHEMA DI COMUNICAZIONE

- DA U.E. A eNODE B (ACCESSO RADIO)

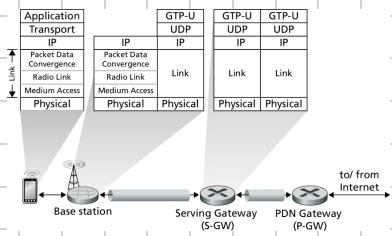
Si usano F.D.R. o T.D.M su più timeslots da 0.5 ms

- DA eNODE B A PD-GW

I datagrammi sono incapsulati con il protocollo

GPRS fino al S-GW che li re-incapsula fino al

PD-GW usando UDP.



● 5G

Evoluzione del 4G non retro-compatibile fisicamente che si basa su onde millimetriche.

+ Alta velocità - bassa copertura - suscettibili a interferenze meteo

L26 >

-> **Mobility:** Intro

-> Come sono classificabili le reti di mobility?

-> Identificazione:

-> Come puo' essere identificato un host in mobilità?

-> Dove viene salvata l'identificazione?

-> Un host necessita di altre identificazioni?

-> Se si, quali?

-> Se si, a cosa servono?

-> **Mobility:** Instradamento

-> Instradamento indiretto:

-> Spiega le fasi che lo caratterizzano.

-> Quali sono i suoi vantaggi e svantaggi?

-> Qual'è il ruolo del PDG-GW?

-> Instradamento diretto:

-> Qual'è il problema che prova a risolvere?

-> Quali sono i suoi vantaggi e svantaggi?

-> **Mobile IP:**

-> Per cosa è stato creato?

-> È utilizzato?

-> Quali sono le sue similitudini con le reti cellulari?

-> Quali sono le sue funzionalità?

-> **Impatti su upper layer:**

-> Quali sono gli impatti della mobilità sull'upper layer?

MOBILITY > INTRO

CLASSIFICAZIONI NETWORK DI MOBILITY

HOME NETWORK:

Rete di appartenenza dell'U.E., in cui viene censito dall'H.S.S.

FOREIGN / VISITED NETWORK

Rete non-home a cui l'U.E. si connette in mobilità.

La mobility dell'host richiede poter "aggiustare" dinamicamente l'addressing.

IDENTIFICAZIONE:

L'Host necessita di essere identificato nella **Home Network**.

- IMSI: Nel caso di reti 4G/5G
- IP Permanente: Nel caso di Mobile IP.

Questa informazione è storizzata nel H.S.S.

In aggiunta, necessita di un IP nel **Visited Network**, come:

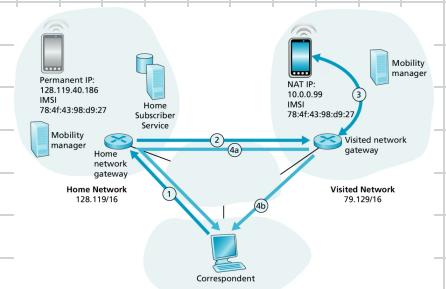
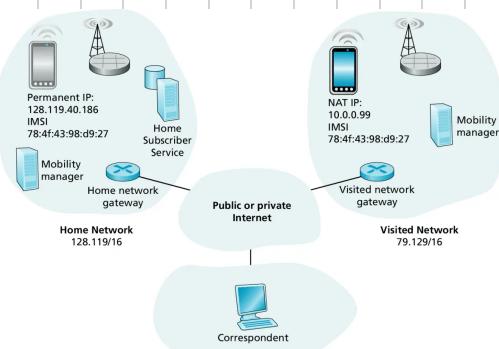
- L' IP permanente dell' Home Network.
- Un IP del Foreign Network.
- Un IP assegnato tramite NAT (privato)

MOBILITY > INSTRADAMENTO

INDIRETTO:

È un approccio trasparente agli hosts comunicanti (mantiene le connessioni aperte):

1. Il corrispondente invia il pacchetto verso l'indirizzo permanente dell'host. (1)
2. Il PDN-GW intercetta il pacchetto e determina con l'**H.S.S.** se l'host è in una visited network. (2)



3. Se sì, il PDN-GW effettua il routing verso il gateway della visited network **incapsolando** il datagramma del corrispondente (Tunnelling) in uno più grande. (2)

4. Il PDN-GW della visited network **de-capsulerà** il datagramma e lo invierà al foreign Host. (2)

In fase di risposta si può scegliere fra:

A. Inviare il datagramma indietro con **tunneling** verso la Home Netw.

B. Inviare il ~~//~~ direttamente al corrispondente

- TRIANGLE ROUTING:

I pacchetti viaggiano fino alla home network **anche** se ci sono percorsi più efficienti [ES: Host e corrispondente su stessa rete]

• DIRETTO

È un approccio che **resolve** il problema del **triangle routing** ma richiede più complessità

1. Il corrispondente interroga la Home Net.

sulla locazione dell'Host (Tramite l'HSS). (1-2)

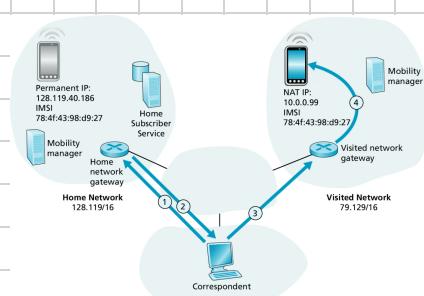
2. Ottenuto l'indirizzo nella foreign network, il corrispondente dialoga **direttamente** con l' Host (3)

Presenta però degli svantaggi:

- Necessita di un protocollo di comunicazione per interrogare l'HSS
- ~~//~~ di un modo di gestione se l'Host cambia rete durante la comunicazione con il corrispondente.

MOBILE IP >

È un protocollo creato 20 anni fa per gestire la mobilità che però non è stato usato per via della già diffusione delle tecnologie mobili cellulari.



SIMILITUDINI:

Usa un sistema di Home / Visited networks come in 4G/LTE

- HOME AGENT: Equivalente al H.S.S. della home network
- FOREIGN : \equiv al N.M.E. della Foreign \equiv .

FUNZIONALITÀ:

Offre protocolli per:

- AGENT DISCOVERY: Foreign agent fa advertising e registrazione agli host connessi.

INDIRECT ROUTING:

- REGISTRATION: Degli IP secondari della foreign network (care of address) nell'Home agent.

MOBILITY > IMPATTI SU UPPER LAYER.

Logicamente sono minimi visto che il net. level offre best effort.

A livello performance, ci possono essere perdite maggiori, che vanno poi a influire maggiormente nel congestion control di TCP.