

Reti I

Daniel Biasiotto

May 31, 2022

CONTENTS

1	Info Corso	1
2	Teoria	2
2.1	Introduzione alle Reti di Calcolatori	2
2.1.1	Componenti	2
2.1.2	Protocolli	7
2.1.3	Internet Standards	7
2.1.4	Packet Delay	7
2.1.5	Throughput	8
2.1.6	Servizi	8
2.1.7	Storia	8
2.2	Livelli	10
2.2.1	Livello Applicativo	10
2.2.2	Livello di Trasporto	21
2.2.3	Livello di Rete	27
2.2.4	Livello di Collegamento	41
2.2.5	Incapsulamento	52
2.3	Sicurezza	52
2.4	Reti Wireless	53
2.4.1	Collegamento Fisico	54
2.4.2	CDMA	54
2.4.3	Protocolli	55
2.4.4	Cellulari	57
2.4.5	Mobilità	58

- [PDF Version](#)

1 INFO CORSO

- Prof:

- Marco Botta
- Orari:
 - Mar 16-18
 - Mer 16-18
 - Gio 11-13 (in alternativa a **ProgIII**)
- Testo:
 - **Computer Networking - A Top-Down Approach**

2 TEORIA

2.1 Introduzione alle Reti di Calcolatori

Internet come esempio delle reti

2.1.1 Componenti

- Computing Devices connessi
 - hosts / sistemi terminali
 - * alla periferia della rete
 - * eseguono degli applicativi
 - applicazioni di rete
 - links
 - * wireless
 - * wired
 - router
 - * collegati tra di loro
 - * collegati agli hosts
 - permettono a questi di comunicare a grandi distanze
 - packet switching
- Gerarchia
 - Internet come rete di reti
 - home Network
 - * Regional ISP
 - Global ISP
 - mobile Network

* Institutional Network

In sostanza:

- hosts: clients, servers / dispositivi periferici alla rete / edge networks
 - in prevalenza wireless
 - access network punto di accesso
 - * per connettere terminali si utilizzano *router di frontiera*
 - * reti di accesso residenziale/istituzionale/mobile
- routers in profondità nella rete / core network
 - in prevalenza wired

ACCESSO

Digital Subscriber Line

DSL Utilizza linee telefoniche esistenti

- splitter
 - dati vanno su Internet
 - voce va sulla rete telefonica

La linea é asincrona:

- Up < 1Mbps
- Down < 10Mbps
 - poiché questa é la piú utilizzata generalmente

Collegato direttamente alla Centrale

Cable Network

Via cavo, utilizzata dagli anglosassoni

- cavo condiviso da vari utenti
- cable modem
 - collegato da uno splitter
 - * dati vanno su Internet
 - * segnale televisivo va su TV
 - * vengono utilizzate tecniche di multiplexing
- Hybrid fiber coax
 - Up < 2Mbps
 - Down < 30Mbps

A differenza della DSL non ha un collegamento diretto al centrale

Enterprise Access Network

Ethernet, sono dei Router che fanno parte della rete internet del ISP

- institutional router
 - Ethernet switch
 - * connessioni singole
- Velocità: 10Mbps, 100Mbps, 1Gbps, 10Gbps

Wireless Access Network

- LAN:
 - 100ft
 - 802.11 b/g: 11Mbps, 54Mbps
- Wide-Area
 - telco operator, 10km
 - 1 - 10 Mbps
 - * 3G, 4G, LTE

HOST Invio di pacchetti di dati tra gli Host pacchetti di bit di lunghezza L, trasmissione a velocità R

- packets
- link bandwidth

Packet Transmission Delay = $\frac{L}{R}$

MEZZI TRASMISSIVI

- guidati
 - il segnale segue un percorso ben preciso
 - cavi
 - * Twisted Pair (TP)
 - 2 cavi di rame intrecciati
 - * Coaxial Cable
 - 2 cavi di rame concentrici
 - maglia di rame intorno per schermare
 - cavo interno bidirezionale banda larga
 - * Fiber Optic
 - fibre di vetro all'interno della quale passa la luce

- il segnale é luminoso, ogni bit é un impulso luminoso
- molto flessibili
- molto veloci, immune alle interferenze elettromagnetiche
- tasso di errore molto basso
- non guidati
 - onde radio
 - * segnali trasportati nello spettro elettromagnetico
 - * possono essere riflessi, ostruiti, interferiti
 - * Microonde terrestri
 - < 45Mbps
 - * LAN WiFi
 - < 11 - 54Mbps
 - * Wide-Area
 - 3G, 4G
 - * satellitare
 - delay per la distanza con il satellite
 - da Kbps a Mbps

NUCLEO DELLA RETE Packet Switching preferito perché permette a piú utenti di usare la rete contemporaneamente

- questo in quanto é bassa la probabilità che tutti gli utenti siano attivi contemporaneamente

PACKET SWITCHING Router Interconnessi che comunicano usando il packet switching

- ogni pacchetto ha lo stesso formato e dimensione, semplici da smistare verso la destinazione
- ogni pacchetto usa l'intera capacità del collegamento per essere inviato

Store-And-Forward

- il router deve ricevere almeno parte del pacchetto, supponiamo tutto
- solo allora puó decidere su quale collegamento in uscita trasmetterlo

Queuing and Loss

- in uscita solo un pacchetto può passare per volta
 - si formano delle code nel buffer del router
 - se la coda è piena un nuovo pacchetto viene scartato
- Funzioni principali del Nucleo
 - routing - instradamento
 - * Routing Algorithm
 - determina il cammino che i pacchetti dovranno seguire
 - * Local Forwarding Table
 - forwarding - inoltra
 - * copia sull'uscita scelta il pacchetto

Piú semplice del circuit switching, i router non devono attivarsi per creare la connessione, ma condividendo le risorse possono crearsi congestioni

- può essere garantita banda ad alcune app audio/video per ottenere del comportamento simile al circuit switching

Circuit Switching

Creare un collegamento diretto tra mittente e destinatario C'è una ridondanza di collegamenti tra i router

- a seguito di una richiesta i router aprono e chiudono i circuiti per creare una linea diretta di circuiti
- con questo schema non ci sono risorse condivise a differenza del Packet Switching
 - i pacchetti andranno alla massima velocità possibile
 - un circuito non utilizzato può essere considerato uno spreco

1. FDM
2. TDM

2.1.2 *Protocolli*

Skype, TCP, IP, HTTP, 802.11 Definiscono:

- formato, ordine di messaggi ricevuti e mandati
- azioni intraprese su invio e ricezione di messaggi

2.1.3 *Internet Standards*

- RFC: Request for comments
- IETF: Internet Engineering Task Force
 - ente che valuta e accetta standard di comunicazione

2.1.4 *Packet Delay*

$$d_{\text{nodal}} = d_{\text{proc}} + d_{\text{queue}} + d_{\text{trans}} + d_{\text{prop}}$$

- processing
 - dipende dalla congestione
- transmission
 - dipende dalla velocità
 - L/R
- propagation
 - d/s
 - d: lunghezza del collegamento
 - s: velocità di propagazione nel medium
 - * $2e8$ m/sec

Se il buffer è pieno il pacchetto è perso (*packet drop*) $\frac{L \cdot \alpha}{R} =$ traffic intensity α è la quantità media di pacchetti in entrata quando questa frazione supera 1 i bit in entrata superano la capacità trasmissiva, i buffer si vanno a riempire

PACKET LOSS Il buffer (coda) è piena, un pacchetto è perso, viene comunicato al router precedente/alla sorgente/viene ignorato

2.1.5 *Throughput*

due casi:

1. $R_S < R_C$
2. $R_S > R_C$

Throughput, la quantità minima

- compreso il *bottleneck link*

2.1.6 *Servizi*

Internet può essere definito come infrastruttura di servizio ad applicativi

- che ne hanno bisogno per comunicare tra loro

Internet può anche fornire un'interfaccia di programmazione utile alla comunicazione

- Generalmente:
 - server
 - * fornisce un servizio
 - client
 - * accede a servizi

Ma la divisione non è netta, ogni Computer può essere entrambi
L'ISP fornisce una certa bandwidth

- che dipende anche dal proprio PC, che potrebbe fare bottleneck

2.1.7 *Storia*

- 1961: Kleinrock
 - queuing theory dimostra l'efficacia del packet-switching
- 1964: Baran
 - packet-switching in reti militari
- 1967: ARPAnet
 - dell'Advanced Research Project Agency
- 1969

- attivazione primo nodo ARPAnet
- 1972
 - prima vera rete ARPAnet, posta elettronica
- 1970: ALOHAnet
- 1974: Cerf and Kahn
 - architettura di interconnessione delle reti
 - principi alla base dell'architettura odierna
 - * minimalismo
 - * autonomia
 - * controllo decentralizzato
 - * best effort service model
 - * dispositivi stateless
- 1976: Xerox
 - Ethernet
- 1979: ARPAnet ha 200 nodi

Proliferano Reti e Protocolli:

- 1983: TCP/IP
- 1982: smtp e-mail
- 1983: DNS
- 1985: ftp
- 1988: controllo congestioni TCP

Commercializzazione e World Wide Web

- 1990~: ARPAnet decommissionata
- 1990~: Web
 - Berners-Lee
 - * HTTP, HTML
- 1994: Mosaic, poi Netscape

2.2 Livelli

2.2.1 Livello Applicativo

Applicazioni su terminali, permettono uno sviluppo e propagazione software molto veloce

- il software non si occupa dei dettagli implementativi della comunicazione web

CONCETTI DELLE IMPLEMENTAZIONI Esistono strutture diverse per le applicazioni

- client-server
 - server - attende richieste
 - * host sempre acceso
 - * IP permanente
 - * data centers
 - client - invia richieste
 - * comunicano con il server
 - * può essere connesso periodicamente
 - * può avere IP dinamico
 - * non comunicano direttamente tra loro
- peer-to-peer
 - non esiste un server sempre attivo
 - i peer possono comunicare direttamente
 - i peer richiedono servizio ad altri peer che li forniscono
 - auto-scalabile

I processi inviano/ricevono messaggi attraverso i socket

- analogo ad una porta
- il percorso e il trasporto é lasciato ai livelli sottostanti

Per ricevere i messaggi i processi devono avere un identificatore

- l'host ha un IP unico, ma non basta
 - possono esserci tanti processi in esecuzione
- IP-host + port number
 - HTTP server: 80
 - mail server: 25

PROTOCOLLI DI LIVELLO APPLICATIVO Definiscono

- tipo dei messaggi
- sintassi dei messaggi
- semantica dei messaggi
- regole per quando si inviano messaggi e si risponde

Due tipologie

- open protocols
 - RFC liberamente consultabili
 - permettono interoperabilità
- proprietary protocols

Integrità dati

Alcune applicazioni non necessitano dati al 100% corretti Altre necessitano della completa integrità dei dati

Tempi di comunicazione

Alcune applicazioni necessitano una certa temporizzazione, delay basso

Throughput

Alcune applicazioni necessitano un minimo throughput da mantenere per funzionare

- multimedia
- a differenza di file-transfer
 - elastic app

Sicurezza

Criptazione dei dati, integrità dei dati

TCP

- reliable transport
- flow control
- congestion control
- no
 - timing
 - security
 - minimum throughput
- connection-oriented

UDP

- unreliable data transfer
- no
 - reliability
 - flow control
 - timing
 - security

Non fornisce servizi particolari, é utilizzato per esempio da applicazioni multimediali

- permette di inviare dati alla stesso velocità a cui il mittente li può inviare

HTTP

HyperText Transfer Protocol

- pagine = insieme di oggetti
- pagine che hanno riferimenti ad altri oggetti
 - identificati URL
- client: browser
- server: web server

Utilizzando TCP

- lato client inizializza connessione creando socket su client e connettendosi alla porta 80 sul server

HTTP é *stateless*

- non mantiene informazioni riguardo le passate connessioni
- questo perché un protocollo con stato é molto complesso
- *non-persistent*
 - al massimo 1 oggetto viene inviato su TCP
 - * poi si chiude
 - si deve aprire una nuova connessione per ogni UL/DL
 - RTT tempo di andata e ritorno per dati dal client al server
 - * può essere calcolato dal client con questa definizione
- *persistent*
 - viene mantenuta la stessa connessione TCP per un periodo
 - può velocizzare leggermente la comunicazione
 - 1.0
- Metodi
 - POST
 - * web page include input
 - URL
- differenze versioni
 - 1.0
 - * GET
 - * POST
 - * HEAD
 - 1.1
 - * precedenti
 - * PUT
 - * DELETE

1. Status Codes

- 200 OK
- 301 Moved Permanently
- 400 Bad Request
- 404 Not Found
- 505 HTTP Version Not Supported

2. Cookies Dato che il protocollo é *stateless* i cookies sono utilizzati per memorizzare alcune informazioni

- 4 componenti
 - header HTTP response
 - header HTTP request
 - cookies mantenuti sulla memoria del browser
 - DB backend sito Web

Utilizzati per

- mantenere autorizzazioni
- carrelli della spesa
- pubblicità targettizzata
- sessione Web utente (email)

3. Web Caches Per fornire all'utente ciò che richiede senza interagire direttamente con il server d'origine

- una richiesta già fornita può essere risolta da un *proxy server cache*
 - solitamente installati dalle ISP
 - riduce il carico sul link di accesso
 - * secondo un suo *hit rate*
- lo stesso browser inserisce gli oggetti ricevuti in cache

4. Conditional GET Per controllare che gli oggetti ricevuti siano aggiornati

- le cache fanno C.GET al server
 - la risposta non contiene nessun oggetto se la versione in cache sia aggiornata

FTP

File Transfer Protocol

- TCP, per trasferimento affidabile
- client-server
- porta 20-21
- Richiede autenticazione
 - primo TCP - client può navigare il filesystem remoto
 - * chiusa alla fine della comunicazione
 - secondo TCP - dopo file transfer command il server apre connessione (porta 20)
 - * chiusa alla fine della trasmissione del file
- control connection: *out of band*
 - 2 canali diversi
- il server FTP mantiene lo stato
 - directory corrente
 - autenticazione utente

1. Comandi e Codici Comandi

- USER username
- PASS password
- RETR filename
- STOR filename
- CD directory

Codici di ritorno

- 331 Username OK, password required
- 125 data connection already open
- 425 Can't open data connection
- 452 Error writing file

SMTP

Simple Mail Transfer Protocol 3 componenti

- user agents
 - client
 - interfaccia utente
- mail servers
 - i messaggi in uscita e in entrata vengono memorizzati qui
- SMTP
 - utilizzato nella comunicazione diretta tra i mail server, o dai user agents ai server

Specifiche:

- porta 25
- trasferimento diretto dei messaggi tra i server
- 3 fasi
 - handshake
 - transfer
 - closure
- comandi/risposte
 - ASCII
 - Status code & frase descrittiva
- messaggi in ASCII 7-bit
- connessioni persistenti
- protocollo di tipo *push*
 - invia dati al server, al contrario di HTTP
- oggetti multipli fanno parte dello stesso messaggio
 - mentre HTTP incapsula ogni oggetto all'interno di una risposta ognuno

POP₃

- authorization phase
 - user, pass
 - OK, ERR
- transaction phase
 - list, retr, dele, quit

DNS

Domain Name System *protocollo di Livello Applicativo*

- Internet hosts router
 - IP address 32 bit
 - nome simbolico leggibile
- DNS si occupa di mappare IP a nome e viceversa

1. Specifiche

- database distribuito
- host e name server comunicano per risolvere i nomi in IP
- é implementato come Application-Layer
 - la complessità é lasciata ai sistemi terminali
 - * se fosse centralizzato sarebbe l'unico punto di fallimento, database singolo e lontano, grande traffico, manutenzione complessa e costosa
- distribuisce il carico
 - indirizza il client che fa la richiesta verso l'indirizzo IP meno carico di richieste tra quelli disponibili

Il sistema é distribuito e gerarchico

- com *DNS servers*
 - yahoo.com *DNS serves*
 - ...
- org *DNS servers*
 - pbs.org *DNS servers*

- edu *DNS servers*
 - poly.edu *DNS servers*

2. Gerarchia

- a) Root I server DNS radici sono 13, in tutto il mondo
 - interrogati solo se uno dei server sottostanti non riesce a risolvere il nome
- b) TLD Top Level Domain com, org, net, edu, jobs, uk, it, fr Educause e Network Solution gestiscono questi domini
- c) Authorative DNS DNS propri delle organizzazioni pubbliche e private
- d) Local DNS Non appartengono strettamente alla gerarchia
 - ogni ISP ne ha uno
 - quando l'host fa una query questa é inviata a questo DNS
 - gestito localmente
 - se non può risolvere l'indirizzo agisce come proxy e risale la gerarchia
 - * la query può essere *ricorsiva* o *itecativa*

3. Caching Una volta risolto un indirizzo il server lo memorizza

- timeout, per evitare associazioni obsolete - TTL
 - *time to leave*
- tipicamente la cache é mantenuta nei DNS locali

4. Resource Records RR (name, value, type, ttl)

- tipi
 - A
 - * name = hostname
 - * value = IP
 - CNAME
 - * name = sinonimo
 - * value = hostname / nome canonico
 - NS
 - * name = domain

- * value = hostname del Authorative DNS

- MX

- * name = nome

- * value = mailserver

5. Messaggi

- header
 - identification
 - flags
 - * query or reply
 - * recursive or not
 - * recursion available
 - * reply authoritative
- questions
 - name, type
- answers
 - RRs
- authority
- info

ARCHITETTURE

Client-Server

P2P

Peer to Peer Non esiste un server sempre attivo
Utilizzato in

- condivisione di file
 - BitTorrent
 - * file diviso in *chunks* dal server e distribuiti in rete
 - * i peer condividono tra loro (torrent)
 - * **tracker**
 - tiene traccia dei *chunks* dei peer
 - registra i peer
 - * **tit-for-tat**

- ci si scambia *chunks* dai peer piú vicini, piú a contatto
- ogni 30 secondi si selezionano peer random

- VoIP
- streaming

Qualunque Peer é un pari, ognuno di essi puó condividere risorse

1. Skype Inerentemente P2P Server:

- gestisce login
- mette in contatto i peer

Clients

- mappati sui SuperNodi
 - username -> IP

Peer riflettori - *relays*

- I NAT non permettono connessione diretta tra i clients
- i supernodi fanno da *relay*
 - i supernodi tra loro comunicano
 - aprono una connessione tra i dispositivi

PROGRAMMAZIONE SOCKET

- Socket - operato dallo sviluppatore
 - porta tra processo e protocollo di trasporto end-to-end
- TCP - operato dal OS
 - buffer
 - variabili
- UDP
 - non c'è connessione tra client e server
 - * questi si scambiano solo messaggi
 - i dati possono perdersi o essere consegnati in ordine diverso a quello di invio

2.2.2 Livello di Trasporto

Comunicazione logica tra processi

- affidabile, consegna ordinata
 - TCP
- non affidabile, consegna disordinata
 - UDP

MULTIPLEXING

- Multiplexing mittente
 - aggiunge transport header
- Demultiplexing ricevente
 - riceve *IP datagramma*
 - con IP mittente e IP destinatario
 - con numero di porta mittente e destinatario

TCP

- 4-tupla
 - source IP
 - source port number
 - dest IP
 - dest port number

Caratteristiche

- *point-to-point*
- *reliable*, in-order byte stream
- *full duplex data*
- *pipelined*
 - congestion e flow control impostati a window size
 - ACK cumulativi

* del pacchetto che si aspetta di ricevere

- *connection-oriented*
 - handshaking
- *flow controlled*

Segmento

- campi da 32 bit
 - source port # | dest port #
 - sequence #
 - ACK #
 - head len | not used | U | A | P | R | S | F | receive window
 - checksum | urg data pointer
- campi a lunghezza variabile
 - options
 - application

Timeout

- piú lungo del RTT - *Round Trip Time*
 - ma puó variare
- corto
 - trasmissioni non necessarie
- lungo
 - trasmissione poco reattiva a packet-loss
- Si stima RTT
 - tempo dalla trasmissione alla ricezione ACK
 - si fa una media dei Sample
 - * $RTT_{est} = (1 - \alpha) \cdot RTT_{est} + \alpha \cdot Sample$

- dove solitamente $\alpha = 0.125$

- * $DEV_{RTT} = (1 - \beta) \cdot DEV_{RTT} + \beta \cdot |\text{Sample} - RTT_{est}|$

- dove solitamente $\beta = 0.25$

- marginine di sicurezza

- $\text{TimeoutInterval} = RTT_{est} + 4 \cdot DEV_{RTT}$

- timer impostato sul pacchetto piú vecchio di cui non si é ricevuto ACK

ACK

- vari scenari per ridurre il numero di ACK

Fast Retransmit

- 3 ACK duplicati indicano che probabilmente un segmento é andato perso
- non aspettare il timer ma ritrasmetti immediatamente il segmento *unacked*

Connection Management

- handshake
 - si decide di stabilere la connessione
 - si decidono i parametri di comunicazione
- socket buffer, variabile
 - comunicato dal ricevente

Listen → SYN sent → Established Listen → SYN received → Established

1. Congestion Control TCP é un protocollo *fair* rispetto alle connessioni e le loro capacità trasmissive Troppe sorgenti che inviano dati ad una velocità superiore a quella gestibile dalla rete
 - pacchetti perse per buffer overflow ai router
 - lunghi ritardi in coda ai buffer dei router

Con conoscenza perfetta il mittente invierebbe solo quando il router ha spazio libero in buffer, questo ovviamente non può avvenire. Anche se si sapesse prima che il pacchetto è perso per buffer pieno il mittente reinvia. Le ritrasmissioni sono il prezzo da pagare per avere un buon throughput

Due approcci:

- *end-end*
 - congestione inferita dalla perdita e ritardo osservati dai terminali
 - * cambiando la finestra di trasmissione cwnd
 - usato da TPC
 - * *additive increase multiplicative decrease*
 - cresce linearmente, limitata dividendo per 2
 - * mittente incrementa cwnd fino a quando rileva perdita
 - * *slow start*
 - fino alla prima perdita aumenta cwnd esponenzialmente
 - * reazione alla perdita
 - timeout
 - finestra di trasmissione torna a 1
 - *slow start* fino a threshold
 - 3 ACK duplicati (uguale al timeout in TCP Tahoe)
 - finestra di trasmissione dimezzata (TCP RENO)
- *network-assisted*
 - router danno feedback ai terminali
 - bit che indica congestione
 - esplicita una frequenza di trasmissione per il mittente
 - ATM ABR
 - * servizio elastico
 - se il cammino è congestionato il mittente viene limitato

- se il cammino é libero il mittente viene avvantaggiato

- * celle Resource Manager

- mandate assieme alle celle dei dati
- contengono informazioni sulla congestione
- restituiti al mittente dal ricevente con i bit intatti

Le app di multimedia non usano TCP per evitare il throttling dovuto al congestion control, tollerano il packet loss.

UDP User Datagram Protocol

- bare bones
- best effort
 - i segmenti possono essere persi
 - consegna disordinata
- **connectionless**
 - niente handshaking
 - ogni segmento é gestito indipendentemente
- usi
 - streaming
 - DNS
 - SNMP

Non avendo connessione iniziale é piú veloce, non ha limiti di congestion control, header piccoli.

- gestione errori
 - UDP checksum
 - * mittente e destinatario calcolano la checksum e la confrontano

RDT Reliable Data Transfer

- 1.0
 - channel sottostante perfettamente affidabile
 - FSM separate per sender / receiver
- 2.0 - *errors*
 - channel sottostante puo' invertire bit
 - * checksum
 - ACK
 - * receiver comunica al sender OK
 - NAK
 - * receiver comunica al sender che si hanno errori
 - * sender ritrasmette
- 2.1
 - se ACK o NAK corrotti
 - * ritrasmesso il pacchetto
 - per gestire i duplicati sender aggiunge numero di sequenza
 - * 0 0 1
- 2.2
 - stessa funzionalità ma senza NAK
 - ACK dell'ultimo pacchetto ricevuto OK invece di NAK
- 3.0 - *errors and loss*
 - il canale sottostante puó anche perdere pacchetti
 - implementiamo un'attesa ragionevole
 - * dopo di che il mittente se non ha ancora ricevuto ACK ritrasmette
 - * i ritardi inducono del lavoro in piú con delle sovrapposizioni di invio e risposta

Performance

3.0 é corretto, le performance sono problematiche

- il protocollo limita l'uso delle risorse fisiche disponibili

Il protocollo é molto limitato dal RTT in quanto si deve stare in attesa del ACK di risposta per poter procedere

PIPELINING Per risolvere il problema di performance del RDT si continuano a trasmettere pacchetti anche durante l'attesa dell'ACK Ci sono due forme generiche di pipelined protocols:

- Go-Back-N
 - sender invia fino a N pacchetti unacked
 - * c'è una finestra di grandezza N tra tutti i pacchetti comprendente:
 1. pacchetti inviati, senza ACK
 2. pacchetti disponibili ad essere inviati
 - receiver invia solo ACK cumulativo
 - * non lo invia se c'è un gap
 - * non necessita buffering a questo lato
 - si riceve solo nell'ordine corretto, altrimenti si scarta
 - sender ha un timer per il più vecchio pacchetto unacked
 - * quando scade reinvia tutti i pacchetti unacked
- Selective Repeat
 - sender invia fino a N pacchetti unacked
 - receiver invia ACK singoli
 - sender ha un timer per ciascun pacchetto unacked
 - * reinvia solo quello relativo allo scadere

2.2.3 Livello di Rete

Comunicazione logica tra hosts

DATAGRAM *Internet* Non c'è setup di chiamata al livello di rete, i router non mantengono stati sulle connessioni.

- non esiste il concetto di connessione a livello di rete
- pacchetti inviati usando l'indirizzo di destinazione

Caratteristiche:

- scambio di dati tra computer
 - servizio elastico, timing variabile
- connessioni diverse tra terminali
 - servizio poco uniforme
- terminali intelligenti
 - semplice nella rete, complesso ai terminali

VIRTUAL CIRCUIT *ATM* Consiste in:

- path
- VC number
 - pacchetti appartenenti a VC portano il suo numero invece che l'indirizzo destinazione
- voci delle *forwarding tables*
- signalling protocols
 - setup, mantenimento e teardown VC
 - in ATM, frame-delay, X.25
 - non usato nell'internet odierno

I router VC mantengono informazioni sullo stato della connessione.
Tecnologia evoluta dalla telefonia

- terminali semplici
 - complessità all'interno della rete

ARCHITETTURA ROUTER Funzioni principali:

- routing algorithms / protocol
 - *routing processor*
- forwarding datagrams da incoming a outgoing
 - *high-speed switching fabric*
 - * switching-rate
 - N multipli del rapporto input/output
 - * tipologie
 - memory
 - prima generazione
 - 1 pacchetto alla volta
 - computer classico, switching sotto diretto controllo della CPU
 - bus
 - 1 pacchetto alla volta
 - crossbar
 - piú pacchetti per volta
- input
 - line termination
 - * *physical layer*
 - link layer protocol
 - * *data link layer*
 - lookup, forwarding, queueing
 - * datagram dest → lookup con forwarding table
 - * *queuing* per sovrapposizione di output, per Head-of-the-Line blocking

- output
 - datagram buffer, queueing
 - * *scheduling discipline* sceglie datagrammi in coda
 - * *buffering* avviene anche con uno switching veloce per via dei pacchetti che vanno allo stesso output
 - link layer protocol
 - line termination
- buffer
 - dimensione approssimata a $\frac{RTT \cdot C}{\sqrt{N}}$
 - * C link capacity
 - * N numero di flussi

INTERNET PROTOCOL IP Protocolli:

- routing
- IP
 - altri 20B di intestazioni
 - complessivamente 40B di overhead (TCP + IP) per ogni pacchetto
- ICMP

IP Fragmentation

Diversi collegamenti all'interno della rete hanno MTU diversi

- Max. transfer size

Datagrammi di grandi dimensione frammentati all'interno della rete

- riassembleti alla destinazione
- utilizzati i campi dell'intestazione IP per riassemble i ordine corretto
 - *fragflag*
 - *offset*
 - * su 13 bit
 - offset di 8B sul file (moltiplicare per 8 per posizione esatta)

Addressing

identificatore a 32-bit per host, interfaccia del router

- un IP per interfaccia
 - gestiti dall'ICANN
 - * Internet Corporation for Assigned Names and Numbers
 - IP assegnati dinamicamente nella sottorete con DHCP
 - * client-server
 - * il protocollo permette di utilizzare stessi indirizzi in tempi diversi a host diversi
 - * DHCP discover broadcast a tutti
 - offer
 - request
 - ACK
- Classless InterDomain Routing
 - CIDR
 - porzione di sottorete dell'indirizzo
 - formato:
 - * a.b.c.d/x
 - x # bit nella porzione di sottorete dell'indirizzo
- gli ISP prendono carico degli indirizzi di loro dominio e di tutti i pacchetti a loro indirizzati

Network Address Translation

NAT Gli indirizzi, anche se di numero molto grande, non soddisfano la domanda

- in quanto sono assegnati in blocco

Le reti locali utilizzano un solo IP per tutti i dispositivi

- i singoli terminali non sono direttamente raggiungibili
 - livello di sicurezza in più
- si può cambiare ISP senza cambiare IP a tutti i dispositivi in rete locale

Implementazione:

- datagrammi in uscita
 - IP sostituito con NAT
 - porta originale sostituita con una porta assegnata a quel pacchetto in particolare
 - altri pacchetti della stessa connessione riutilizzano sempre la stessa porta assegnata fino a che necessario
- datagrammi in entrata
 - tradotto attraverso la NAT translation table

Controverso:

- i router non dovrebbero modificare le intestazione di livelli superiori, livello di rete e di trasporto
 - il NAT modifica il livello di trasporto
- viola la comunicazione punto-punto tra host
 - questo crea problemi nel P2P ad esempio
 - * risolvibile attraverso *relay*

Per rendere raggiungibile direttamente dall'esterno un dispositivo posto dietro NAT é possibile:

- associare un indirizzo pubblico ad un indirizzo interno direttamente all'interno del Router
- utilizzare il protocollo UPnP
 - Universal Plug and Play
 - automatizza la soluzione statica precedente
- *relay*

ICMP

Protocollo di messaggistica

- utilizzato da host e router per comunicare informazioni a livello di rete
- ping

Messaggi ICMP costituiti da

- tipo
- codice

Utilizzato da traceroute

IPv6

128 bit - 16 Byte Motivazioni principale

- estendere lo spazio degli indirizzi
- migliorare la velocità di elaborazione dell'intestazione
- non più *best-effort* ed *elastica* ma per facilitare il Quality of Service

Formato:

- lunghezza 40B fissa
- frammentazione non permessa
 - aggiunge messaggi "Packet Too Big"
 - * sarà il mittente ad adeguarsi alla capacità trasmissiva del collegamento
- *flow label* identifica pacchetti dello stesso flusso di dati
- non c'è più il *checksum*
- non ci sono più le *options*
 - possono essere inserite al di fuori dell'intestazioni

Per la transizioni viene utilizzato il *tunnelling*

- IPv6 pacchetti trasportati come *payload* all'interno di pacchetti IPv4 attraverso router IPv4

ROUTING ALGORITHMS vedi: **Cammini Minimi**

Astrazione in forma di grafo $G = (N, E)$

- N insieme di *router*
- E insieme di *link*

Costi: $c(x, x')$ - costo link (x, x')

Specifiche:

- global | decentralized
 1. tutti i router hanno topologia completa
 - link state algorithms
 2. i router conoscono solamente i vicini direttamente connessi
 - distance vector algorithms
- static | dynamic
 1. i cammini cambiano lentamente
 2. i cammini cambiano velocemente, l'algoritmo può reagire ai cambiamenti

Link-State Routing

Algoritmo di Dijkstra

- $O(n^2)$
 - esiste anche un costo in quanto i router si devono scambiare necessariamente dei messaggi per avere tutte le informazioni sulla topografia
- esistono implementazioni più efficienti in $O(n \log n)$

I costi dei link sono conosciuti da tutti i router, tutti possono eseguire l'algoritmo di Dijkstra

- producendo la *forwarding table* per quel particolare nodo

Sono possibili **oscillazioni**

- scegliendo un particolare cammino più efficiente fa sì di cambiare il costo stesso del cammino
- aumentando il traffico per quel collegamento

Distance Vector Routing

vedi: **Programmazione Dinamica Bellman-Ford Equation**

$$d_x(y) = \min_v \{c(x, v) + d_v(y)\}$$

dove: \min_v é il minimo calcolato su tutti i vicini v di x
ogni nodo mantiene

- una stima $D_x(y)$ per ogni nodo nella rete
- una stima $D_v(y)$ dei vicini

quando la propria $D_x(y)$ cambia lo si scambia tra vicini

- si ricalcolano le stime
- questo procedimento porta $D_x(y)$ a tendere a $d_x(y)$

L'algoritmo é

- *iterativo*
- *asincrono*
- *distribuito*

L'algoritmo permette:

- una veloce propagazione di cambiamenti positivi della rete
- una lenta propagazione di cambiamenti negativi della rete
 - *count to infinity problem*
 - risolvibile attraverso la *poisoned reverse*

In caso di errori questi si propagano nella rete in quanto i router utilizzano i risultati gli uni degli altri

ROUTING Nella realtà:

- i router non sono tutti identici
- la struttura reale della rete non é piatta

Principali problemi:

- scala
- amministrazione autonoma
 - ogni admin potrebbe voler controllare il proprio routing

Quindi si utilizza un Hierarchical Routing

- collezioni di router

Autonomous Systems

AS

- Un ISP può consistere di più AS
- Router nello stesso AS utilizzano lo stesso protocollo di routing
 - Intra-AS routing algorithm
- Router in AS diversi
 - Inter-AS routing algorithm

Un *gateway router*

- terminale rispetto al suo AS
- connette a router di altri AS

RIP

- advertisement a timer
- se la table cambia si invia advertisement
- utilizza *poison reverse* per evitare ping-pong loops
 - distanza infinita = 16 salti

RIP implementata su livello applicativo (daemon), advertisement attraverso pacchetti UDP

OSPF

Open Shortest Path First

- algoritmo a stato del collegamento
- calcolo dell'instradamento utilizzando Dijkstra
- advertisement direttamente ai vicini
 - *advertisement flooding*
 - propagati per tutto l'AS
- direttamente in datagramma IP
 - non ha bisogno del livello di trasporto

- i messaggi ASPF sono cifrati
- permette piú cammini a costo minimo
- ogni link sono possibili metriche diverse per i costi
- supporto integrato a multicast
- é possibile strutturare grandi domini in livelli gerarchici ulteriori
 - area locale - local
 - area dorsale - backbone

BGP

Border Gateway Protocol

- protocollo tra domini
- eBGP info di raggiungibilitá
- iBGP propaga le info di raggiungibilitá ai router interni alla sottorete
- permette alle sottoreti di informare il resto di Internet della propria esistenza

Si basa sul concetto di *sessione* di messaggi BGP

- *prefix advertisement* tra peers
- si scambiano pacchetti *path vector*
 - si informano le altre AS che cosa é raggiungibile passando dalla propria AS
 - attributi
 - * AS-PATH
 - * NEXT-HOP
- connessioni semi-permanenti TCP
- *sessioni*
 - eBGP
 - iBGP
- *import politics*

– i router di frontiera possono avere politiche differenti per selezionare un *advertisement* piuttosto che un altro per uno stesso prefisso

- * lunghezza del collegamento

- AS-PATH

- *
 -

BROADCAST un singolo nodo trasmette a molti

- duplicazione alla sorgente i pacchetti che invia

- inefficiente

- non é detto che la sorgente conosca tutti gli indirizzi destinatari

- duplicazione all'interno della rete

- *flooding*

- * un nodo che riceve un pacchetto in broadcast lo duplica e invia a tutti i vicini

- * può creare cicli e *broadcast storm*

- *controlled flooding*

- * invia solamente se non già inviato in precedenza

- fatto con i numeri di sequenza (id)

- * o con RPF (Reverse Path Forwarding)

- invio del pacchetto solo se é giunto dal cammino più breve possibile tra nodo e sorgente

- *spanning tree*

- * nessun pacchetto ridondante ricevuto da alcun nodo

- * un albero non ha cicli

- * va costruito

1. selezione di un nodo centrale

- si inviano dei messaggi di join dagli altri nodi

- questi messaggi sono reinviati fino a che non si raggiunge un nodo già inserito nell'albero di distribuzione

MULTICAST sistemi mittenti e sistemi riceventi

- alcuni fanno parte del gruppo multicast altri no
- anche router che non hanno membri multicast possono fare parte della rete multicast se essenziali al collegamento

Approcci alla costruzione dell'albero di distribuzione

- *source-based tree*
 - shortest path trees
 - * **Algoritmo di Dijkstra**
 - RPF
 - * permette il *pruning* su sottoalberi che non contengono membri del multicast
- *group-share tree*
 - minimal spanning (*Steiner*)
 - * albero a costo minimo che connetta tutti i router con membri
 - * problema NP-completo
 - * l'algoritmo é monolitico
 - va rieseguito ogni volta che la rete varia
 - * esistono buone euristiche ma rimane poco usato
 - center-base trees

DVMRP

Distance Vector Multicast Routing Protocol

- *flood and prune*
 - RPF tree basato sulle routing tables costruite comunicando tra i router DVMRP
 - non assume nulla sull'unicast sottostante
 - i router non nel gruppo possono mandare messaggi di pruning upstream

- *soft state*
 - resetta lo stato a intervalli di tempo
- *tunnelling*
 - utilizzato per collegare fisicamente router multicast che sono connessi logicamente
 - collegamenti virtuali
 - * datagrammi multicast inseriti in datagrammi non multicast

PIM

Protocol Independent Multicast

- non dipende dall'algoritmo di routing utilizzato al livello di unicast
- due scenari di distribuzione
 1. *dense*
 - membri posizionati densamente
 - ampiezza di banda più importante
 - i router fanno implicitamente parte della distribuzione
 - * devono chiedere il pruning loro stessi
 - *data-driven* mcast tree (RPF)
 - * *flood and prune*
 - * meccanismo di protocollo per informare i nodi se sono foglie
 2. *sparse*
 - membri largamente sparsi
 - * in reti diverse
 - ampiezza di banda non altrettanto importante
 - l'appartenenza al gruppo va richiesta esplicitamente
 - *receiver-driven* mcast tree (center-based)
 - * i router inoltrano messaggi di *join* verso il *rendezvous point*
 - * i messaggi sono inviati tramite unicast al centro che poi distribuisce

2.2.4 Livello di Collegamento

I protocolli di questo livello lavorano su *frame*, che incapsulano i datagrammi. Il livello tratta di *link* tra *nodi*

- wired
- wireless
- LANs

I protocolli di questo livello si trovano su tutti i nodi della rete

- *network interface card* - NIC
 - scheda di rete

Implementati in parte in hardware, in parte in firmware (controller della scheda), in parte in software

SERVIZI

- *framing*
 - incapsulamento di un datagramma
 - aggiunge header, trailer
 - accesso condiviso se il medium é condiviso
 - MAC address che identificano sorgente e destinazione del *frame*
- *trasferimento dati affidabile*
 - in particolare per i collegamenti con alto tasso di errori
 - * wireless
- *flow control*
- *error detection*
- *error correction*
- *half-duplex - full-duplex*

ERRORI

Detection & Correction

Bit aggiunti al datagramma:

- EDC - Error Detection and Correction bits

I bit sono controllati da ricevente

- possono esserci errori non rilevati anche se raramente
- *Parity checking*
 - parità singola
 - * permette di individuare errori di singoli bit
 - * non molto sicuro ma semplice e veloce
 - parità bidimensionale
 - * permette di individuare e correggere errori di singoli bit
- **Cyclic Redundancy Check**
 - R bit tali che
 - * $\langle D, R \rangle$ divisibile per G
 - permette di individuare fino a r errori di bit consecutivi
 - * questo perché solitamente gli errori si presentano in *burst*
 - $D \cdot 2^r \text{ XOR } R = nG$

PROTOCOLLI AD ACCESSO MULTIPLIO Esistono mezzi *broadcast* condivisi oltre a quelli *point-to-point*

- si verificano interferenze/collisioni se due o più nodi trasmettono allo stesso momento

I protocolli di questo tipo:

- algoritmi distribuiti che determinano quando i nodi possono trasmettere
- le comunicazioni riguardanti la condivisione del canale possono necessitare il canale stesso
 - *in-band channel coordination*

MAC

Medium Access Control Protocols

- *channel partitioning*
 - suddivisione del canale in parti piú piccole
- *random access*
 - il canale non viene suddiviso
 - permette le collisioni
 - quando un nodo ha bisogno del mezzo lo utilizza
 - * vengono individuate le collisioni
 - * si specifica come risolvere la collisione
 - tipicamente con la ritrasmissione ritardata
- *turns*
 - i nodi vanno a turni

1. Channel Partitioning

a) TDMA - channel part Time Division Multiple Access

- accesso diviso su turni
- ad ogni nodo é assegnato uno slot temporale
 - slot non utilizzati vanno sprecati

b) FDMA Frequency Division Multiple Access

- banda divisa in bande piú piccole
 - su frequenze diverse
- ogni nodo é assegnato ad una sotto-banda

2. Random Access

a) ALOHA

- frame di grandezza uguale

- tempo diviso in slot uguali
- i nodi sono sincronizzati
- se 2 piú nodi trasmettono nello stesso slot
 - tutti i nodi registrano la collisione
- quando un nodo riceve un frame, trasmette nello slot successivo
 - se si verifica una collisione si tenta di ritrasmettere nello slot successivo con una certa probabilità p , altrimenti ritenterá allo slot successivo ancora con la stessa probabilità

Quindi:

- prob che un nodo abbia successo in un dato slot, $p(1 - p)^{N-1}$
- prob che un qualsiasi nodo abbia successo, $N \cdot p(1 - p)^{N-1}$
- massima efficienza massimizza questo valore, $1/e = 0.37$
 - nel caso migliore solo il 37% del tempo uno slot viene sfruttato

Nella versione pura, *unslotted*, di ALOHA non c'è sincronizzazione e qualsiasi nodo può trasmettere immediatamente quando necessario

- le collisioni avvengono nelle sovrapposizioni tra le trasmissioni
- le collisioni aumentano

In questa versione:

- prob che un qualsiasi nodo abbia successo, $p \cdot (1 - p)^{N-1} \cdot (1 - p)^{N-1}$
 - $p \cdot (1 - p)^{2(N-1)}$
- massima efficienza $1/e = 0.18$

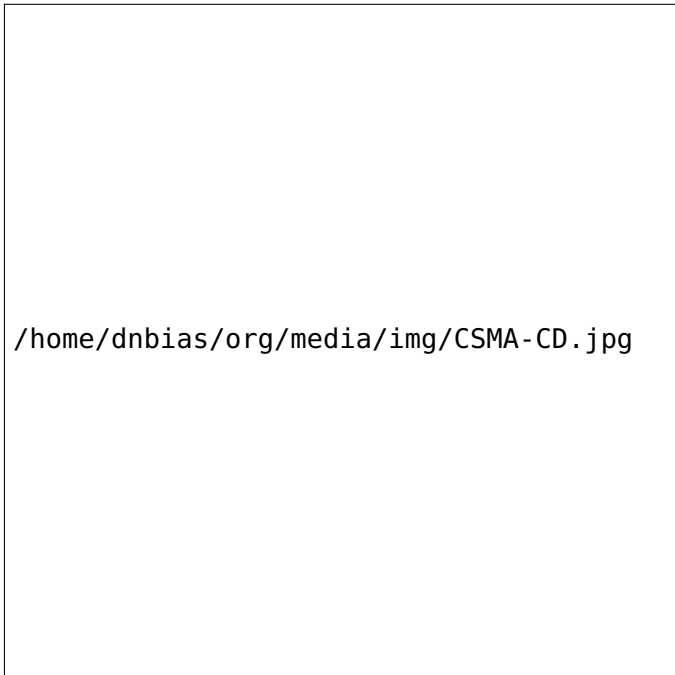
b) CSMA Carrier Sense Multiple Access Non interrompe la comunicazione altrui

- se il canale viene individuato come in *idle*
 - trasmette l'intero frame
- se il canale é *busy*
 - rimanda la trasmissione

A causa del *propagation delay* le collisioni possono ancora accadere

- i nodi potrebbero non accorgersi in tempo che il canale é occupato in realtà
- in caso di collisione tutto il tempo di trasmissione viene sprecato

i. CSMA/CD CSMA Collision Detection



/home/dnbias/org/media/img/CSMA-CD.jpg

In caso di collisioni si individuano velocemente per interrompere la trasmissione

- riduce lo spreco del canale
- Collision Detection

- LAN
 - * si confronta l'intensità del segnale trasmesso e ricevuto
 - * le interferenze creano una variazione di potenza del segnale
 - * protocollo utilizzato attualmente per reti Ethernet

- Wireless
 - * più complesso
 - * il segnale ricevuto solitamente è attenuato mentre la trasmissione è più potente

- efficienza

- t_{prop} max delay di propagazione tra 2 nodi
- t_{trans} tempo di trasmissione di un frame di dimensione massima
- $efficiency = \frac{1}{1 + 5 \frac{t_{prop}}{t_{trans}}}$
 - * aumenta con l'aumentare di t_{trans}
 - * aumenta con il diminuire di t_{prop}

A. Algoritmo Ethernet CSMA/CD

- NIC riceve datagramma, crea frame
- NIC controlla il canale
 - se occupato aspetta
 - se libero trasmette
 - * se non determina disturbi durante la trasmissione suppone che il frame sia stato inviato correttamente
 - * se determina disturbi invia un *jam signal* e interrompe la trasmissione

- reinvia dopo un determinato quantitativo di tempo
- *binary backoff*
- k casuale tra 0 e $2^m - 1$ con m collisioni
- NIC aspetta $k \cdot 512\text{bit}$

3. Turns

a) Polling

- nodo *master* invita i nodi *slave* a trasmettere a turno
- solitamente *slave* "dumb"
- downsides
 - overhead del *polling*
 - latenza
 - singolo punto di fallimento - *master*

b) Token

- *token* di controllo passato da un nodo all'altro in sequenza
 - downsides
 - *token* overhead
 - latenza
 - singolo punto di fallimento - *token*
- * può essere perso

LANS

Addressing

Il nodo destinazione nella rete locale é individuato con gli indirizzi fisici MAC o LAN

- 48 bit scritto in ROM del NIC
 - scheda di rete
 - 6 coppie esadecimali di 4bit ciascuno
 - indirizzo piatto, non cambia in base alla rete cui é connessa a differenza dell'indirizzo IP
- l'indirizzo é univoco (unico per la singola scheda)
 - amministrato da IEEE che assegna porzioni di indirizzi a produttori diversi
 - * garantisce l'unicitá
- utilizzato per trasferire da un'interfaccia ad un altro, stesso IP per quanto riguarda il livello di rete

1. ARP Address Resolution Protocol Passaggio da IP a MAC

- nodo contiene una ARP table
 - associa IP/MAC
 - <IP;MAC;TTL>
 - * *Time to Leave* in quanto il collegamento alla sottorete puó cambiare
- quando si necessita di un MAC si manda la richiesta nella propria rete
 - il nodo cui corrisponde l'IP della richiesta e se questo é il suo risponde con il proprio MAC
 - cosí viene popolata la ARP table

Ethernet

Inizialmente pensato con una gerarchia definita:

1. transiver
2. cavo
3. interfaccia
4. controller

Permettava una velocità tra 10Mbps e 10 Gbps Inizialmente la topologia era di **bus** La topologia attuale é quella a **stella**

- ora con una *switch* al centro
- collegamenti diretti tra nodi e *switch*
- non c'è collisione

I *frame* Ethernet sono gli stessi indipendentemente dalla velocità trasmissiva

- preamble
- destination (MAC)
- source (MAC)
- type
 - solitamente IP ma può essere un qualunque protocollo
- data
- CRC

I *frame* che non sono indirizzati al ricevente sono scartati

- *connectionless*
- *unreliable*
 - non sono inviati ACK
- protocollo MAC: CSMA/CD w/ binary backoff
 - *jam* signal in caso di disturbo
 - * 48 bit
 - * questo quantitativo di bit é utilizzato per attendere un tempo dipendente dalla velocità del collegamento
 - attesa esponenziale
 - * l'adattatore tenta di stimare quanti sono gli adattatori coinvolti

Switches

funzione di store-and-forward dei *frame* Ethernet

- *trasparenti*
 - gli host non sono a conoscenza degli switch
- *plug-and-play*
- *self-learning*

Grazie a questi sono possibili trasmissioni simultanee

- riceve i *frame* e li inserisce in buffer
- sceglie i buffer del collegamento in uscita desiderato per i pacchetti
 - le interfacce sono associate ai nodi raggiungibili tramite una *switch table*
 - * MAC - interfaccia - timestamp TTL
 - * popolata tramite autoapprendimento
 - * ogni volta che un *frame* viene ricevuto viene memorizzata l'associazione tra interfaccia e MAC del mittente
 - * se il destinatario non é all'interno della tabella allora il *frame* viene inviato su tutte le interfacce

· *flood*

- le collisioni non avvengono in quanto pacchetti non si incontrano perché smistati su code diverse dallo switch

Gli switch possono essere collegati tra di loro per creare sottoreti e strutture complesse

VLANS

Permettono di risolvere problemi di privacy e sicurezza delle LAN Porte di uno *switch* raggruppate in modo che un solo *switch* fisico operi come più dispositivi virtuali

Se switch diversi fanno parte di una stessa VLAN questi sono collegati tra loro da porte particolari:

- *trunk port*
 - scambia *frame* aggiungendo degli identificatori
 - protocollo 802.1Q VLAN

VIRTUALIZZAZIONE**MPLS****Multiprotocol Label Switching**

- goal:
 - velocizzare l'inoltro
 - la ricerca dell'IP di destinazione nella tabella di inoltro può essere lenta
 - introdurre delle etichette per velocizzare lo switching all'interno del frame
 - * *header* MPLS
 - nuova tabella di inoltro più efficiente
 - * l'inoltro viene fatto in base all'etichetta e non viene ispezionato l'IP
 - * capacità di bilanciare il carico
 - l'invio ad uno stesso destinatario può essere diverso in base all'etichetta associata al *frame*

Simile ai Circuiti Virtuali

DATA CENTER NETWORKING Da decine a centinaia di *hosts* in prossimità

L'obiettivo è bilanciare il carico ed evitare bottleneck nell'accesso ai dati

1. Server rack
2. TOR switches
3. Tier-2 switches
4. Tier-1 switches
5. Access Router
 - Load Balancer
 - riceve le richieste client esterne
 - dirige il traffico nel data center

– ritorna i risultati ai client esterni

* nasconde l'interno del data center dai client

6. Border Router

Possono essere possibili diversi schemi di connessione tra i livelli di switch per ottimizzare l'accesso

2.2.5 *Incapsulamento*

Ogni livello che si discende si aggiunge una intestazione

2.3 Sicurezza

Non é stata pensata inizialmente con la sicurezza in mente

- facilitare la comunicazione tra ricercatori
 - con trasparenza

Malware:

- virus
- worm
- spyware
- botnet

Attacchi DoS

- Denial of Service
- attaccanti rendono le risorse sul server non disponibili per il traffico reale con moltissime richieste

Packet Sniffing

- con l'accesso ai mezzi trasmissivi (spesso condivisi)
- intercettazione dei pacchetti trasferiti nel percorso compromesso

IP spoofing

- invio di pacchetti con IP falso, rubato

2.4 Reti Wireless

Oggigiorno le connessioni Wireless sono in numero molto maggiore rispetto a quelle cablate

- wireless hosts
 - mobile o meno
- stazioni base
 - funzione di relay tra rete cablata e dispositivi host wireless
 - torri cellulari
 - access points
- wireless link
 - collegamento tra dispositivi e stazioni base
 - anche tra stazioni base come collegamento di dorsale, *backbone link*

Modalità:

- Infrastruttura
 - stazioni base cui si connettono i dispositivi che permette il collegamento alla rete Internet
- ad hoc
 - non c'è una stazione base
 - i nodi trasmettono agli altri nodi
 - i nodi devono organizzare loro stessi una rete
 - * funzioni di *routing* e *forwarding*
- single hop
- multiple hop

2.4.1 Collegamento Fisico

- il segnale radio viene attenuato maggiormente con l'attraversamento dell'etere
- interferenza con altre sorgenti
 - frequenze occupate da altri
 - motori
- il segnale radio viene diffuso in tutte le direzioni e può essere riflesso, producendo cammini differenti, ritardi e sovrapposizioni

SNR - Signal-to-Noise Ratio BER - Bit Error Rate

Questi sono inversamente proporzionali (quasi esponenziale)

- si aumenta la potenza di segnale in funzione del mezzo fisico utilizzato per minimizzare il BER
- un piccolo calo SNR introduce un BER elevato

Se dei terminali sono nascosti questi non possono verificare le trasmissioni l'uno dell'altro

- trasmettono comunque e interferiscono tra loro
- *hidden terminal problem*

2.4.2 CDMA

Code Division Multiple Access

- gli utenti condividono la stessa frequenza
- gli utenti hanno una frequenza di *chipping* per codificare le proprie trasmissioni
- il ricevente può estrarre il segnale utilizzando la frequenza di *chipping*
- segnali ortogonali interferiscono tra loro ma è possibile recuperare i dati originali

2.4.3 *Protocolli*

802.11 Wireless LAN Copertura abbastanza limitata ma ad alta velocità

- esiste una versione a lunga portata: 802.11a,g point-to-point
- 2.4GHz - 2.485GHz
 - 11 canali
 - i canali si sovrappongono, possono avvenire delle interferenze
 - * solo 1, 6, 11 non si sovrappongono
 - per gestire l'aumento di BER si cambia tipo di trasmissione, più lenta ma anche più efficace

Le stazioni base in modalità infrastruttura sono gli AP, Access Points

- trasparente rispetto alla rete

Le *cells* sono i BSS, Basic Service Sets

- in modalità ad hoc contengono solo i terminali

Gli host si associano agli AP

- passive scanning
 - frame beacon trasmessi dall'AP
 - * contiene SSID e MAC dell'access point
- active scanning
 - probe request broadcast
 - probe request frame dall'AP

Accesso Multiplo

Evitare le collisioni Utilizza CSMA

- se si nota una trasmissione in corso si aspetta

Rimangono problemi

- decadimento del segnale
- terminali nascosti

Non é implementata collision detection

- Sender
 - DIFS
 - * timer di attesa
 - se canale *idle* dopo DIFS
 - * trasmetti frame
 - altrimenti
 1. aspetta random backoff time
 2. trasmetti frame
 3. aspetta ACK, se non lo si riceve si riparte da 1.
- Receiver
 - frame ricevuta OK
 - trasmetti ACK dopo SIFS
 - * ACK necessario a causa del problema del terminale nascosto

Capacità ulteriori

- power management
 - nodo si spegne attendendo il prossimo *beacon frame*
 - AP non trasmette nodi a questo nodo

802.15 Infrastruttura *master - slave*

- 10m di diametro
- rimpiazza i cavi per device
 - mouse, tastiere, cuffie
- ad hoc
 - non c'è infrastruttura
- evoluzione del Bluetooth
- 2.4 - 2.5 GHz
- fino a 721 kbps

2.4.4 Cellulari

Basate sul concetto di cella

- *base station*
 - potenza molto alta ma per il resto equivalente agli AP
- *air interface*
- MSC - Mobile Switching Center
 - connette le celle
 - gestisce la mobilità dei device
 - * hand-over da una stazione base ad un'altra

Tecniche di condivisione di banda:

- FDMA / TDMA
 - divide spettro in canali di frequenze
 - divide canali in slot temporali
- CDMA
 - code division multiple access

2G Voice network

1. BSS
 - stazioni base
2. MSC
3. Gateway MSC
4. rete telefonica pubblica, cablata

3G Voice + Data network Divisione tra le due reti per permettere scalabilità

1. BSS
2. radio network controller
 - a) MSC
 - gateway
 - rete telefonica pubblica
 - b) SGSN
 - GGSN
 - Rete Internet

2.4.5 Mobilità

Un dispositivo che si sposta, si connette e disconnette attraverso AP diversi

- la mobilità piú elevata si ha quando il dispositivo pur spostandosi e cambiando AP mantiene una stessa connessione
- home network
 - indirizzo IP permanente
- home agent
 - gestisce le funzioni di mobilità per conto del dispositivo mobile quando quest'ultimo non si trova all'interno della rete domestica
- visited network
- foreign agent

Ci sono diversi approcci possibili

- si lascia la gestione ai *router*
 - non scalabile
 - le tabelle sarebbero ingestibili con milioni di utenti che si spostano

- si lascia la gestione ai terminali
 - *indirect routing*
 - * comunicazione passa per *home agent* poi per *remote*
 - *direct routing*
 - * il corrispondente ottiene l'indirizzo *foreign* del mobile

Il mobile si registra contattando il *foreign agent*

- questo contatta l'*home agent*
- entrambi sanno come contattare l'utente mobile

MOBILE IP

- indirect routing
- agent discovery
 - agent advertisement
 - * foreign/home agents fanno broadcast di messaggi ICMP

CELLULARE

- indirect routing
 - switching center home controlla il proprio registro
 - contatta lo switching center visitato creando la connessione

HANDOFF

1. vecchio BSS informa MSC dell'handoff
 - lista di piú di uno BSS
2. MSC imposta il cammino per il nuovo BSS
3. nuovo BSS alloca canale audio per l'uso
4. nuovo BSS segnala MSC e vecchio BSS
 - pronto
5. vecchio BSS segnala MSC

- pronto
6. mobile nuovo segnale BSS per attivare il nuovo canale
 7. mobile segnala attraverso nuovo BSS a MSC
 - handoff complete
 8. MSC rilascia risorse del vecchio BSS

In caso di MSC diversi l'originale ha funzione di ancora verso un nuovo MSC